INTERNATIONAL ATOMIC ENERGY AGENCY
UNITED NATIONS EDUCATIONAL, SCIENTIFIC AND CULTURAL ORGANIZATION
INTERNATIONAL CENTRE FOR THEORETICAL PHYSICS
I.C.T.P., P.O. BOX 586, 34100 TRIESTE, ITALY, CABLE CENTRATOM TRIESTE

H4.SMR/383 - 13

WORKSHOP ON REMOTE SENSING TECHNIQUES WITH APPLICATIONS TO AGRICULTURE, WATER AND WEATHER RESOURCES

(27 February - 21 March 1989)

TWO-DIMENSIONAL DIGITAL FILTERS

AND

DATA COMPRESSION

V. CAPPELLINI
University of Florence
and
IROE - CNR
Florence
ITALY

Two-Dimensional Digital Filters and Data Compression

V. CAPPELLINI

Dipartimento di Ingegneria Elettronica
University of Florence and IROE-C.N.R.
Florence, Italy

I. INTRODUCTION

The area of digital image processing is of increasing importance and interest due to the impressive advances and innovations in digital methods and technology. In particular one may outline the following aspects: definition of new and efficient algorithms to perform linear and nonlinear operations with great application flexibility and adaptivity; production of fast new image digitizing systems and high-resolution displays; efficient software or hardware implementation capabilities due to the large expansion and evolution of standard computers, minicomputers, array-processors, microprocessors, and high-integration digital circuits (LSI, VLSI), available at a decreasing cost.

Digital image processing methods and techniques are being increasingly applied in several important fields such as communications, radar–sonar systems, remote sensing, biomedicine, office automation, moving-object recognition, and robotics.

High-interest digital operations, which can be performed for image processing, are the following: two-dimensional (2D) digital transformations, 2D digital filtering, local space processing, data reduction or compression, and pattern recognition. Digital transformations, digital filtering, and local space operators can be used to perform smoothing, enhancement, noise reduction, and edge extraction. Data compression operations permit one to reduce a large amount of data representing the images in digital form, solving transmission or storage problems. Pattern recognition is used to extract useful configurations from images for final interpretation and utilization.

In this article 2D digital filtering and data compression operations are mainly described, pointing out their crucial importance for image processing; in particular the joint use of these two digital operations to increase the overall efficiency of image processing is presented. Local space operators are also described as simpler 2D digital filters defined only in the space domain (while the more complex digital filters are defined and designed both in the space and frequency domains).

After a synthetic review summarizing the separate digital operations (2D digital filters, local space operators, data compression) and the presentation of their joint use, some examples of applications to important fields such as communications, remote sensing, biomedicine, and robotics are described.

II. Two-Dimensional Digital Filters

The following aspects regarding 2D digital filters are presented: filter definition and general properties, stability, and designing methods.

A. *Definition of Two-Dimensional Digital Filters and General Properties*

Linear shift-invariant 2D digital filters are defined by 2D difference equations of the type (Cappellini *et al.*, 1978)

$$g(n_1,n_2) = \sum_{R_1}\sum a(k_1,k_2)f(n_1-k_1,n_2-k_2) - \sum_{S_1}\sum b(k_1,k_2)g(n_1-k_1,n_2-k_2) \tag{1}$$

where $f(n_1,n_2)$ are the input data (samples of the input image), $g(n_1,n_2)$ are the output data (samples of the output image), and $a(k_1,k_2)$, $b(k_1,k_2)$ are the coefficients which define the 2D digital filter. R_1 and S_1 are suitable sets, where the indices of the sums can vary to specify different classes of filters.

A first important class of 2D digital filters is obtained when R_1 is defined as the set $0 \le n_1 \le N_1 - 1$, $0 \le n_2 \le N_2 - 1$ and S_1 is the void set. In this case Eq. (1) is reduced to

$$g(n_1,n_2) = \sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-1} a(k_1,k_2)f(n_1-k_1,n_2-k_2) \tag{2}$$

and defines the class of FIR (finite impulse response) 2D digital filters. The difference equation is now a convolution between the input-data matrix and the coefficient matrix, and no feedback of previous output data is present.

In the (z_1,z_2) plane the filter of Eq. (2) is defined by the transfer function

$$H(z_1,z_2) = \sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-2} a(k_1,k_2)z_1^{-k_1}z_2^{-k_2} \tag{3}$$

which can be obtained by applying the 2D z transform to both sides of Eq. (2) and has the form of a bivariate polynomial.

In the above FIR case, assuming that the entire matrix to be processed is available, the sequence of computation is in principle irrelevant, and it is only a matter of computational convenience. On the contrary, in the general case when S_1 is not the void set and some samples $g(n_1,n_2)$ are used in the computation of the current output sample, the sequence of computation is indeed important, because the output samples to be used in (1) have to be available (that is, previously computed or part of the initial conditions). Moreover, the sequence of computation has to be such that the corresponding linear system is *stable*. Therefore, the set S_1 has to be chosen in a suitable way, and a set of initial conditions for the computation have to be defined in such a way to compute any output sample as a function of the previously computed output samples or of the initial conditions.

According to the above considerations, several different sequences of computation can be chosen. The most common one is that corresponding to the so-called quadrant recursive causal filter. In this case $g(n_1,n_2) = 0$ for $n_1 \le 0$ and $n_2 \le 0$, if $f(n_1,n_2) = 0$ for $n_1 \le 0$ and $n_2 \le 0$. The input–output equation can now be written in the form

$$g(n_1,n_2) = \sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-1} a(k_1,k_2)f(n_1-k_1,n_2-k_2) - \sum_{\substack{k_1=0\\k_1+k_2\neq 0}}^{M_1-1}\sum_{k_2=0}^{M_2-1} b(k_1,k_2)g(n_1-k_1,n_2-k_2) \tag{4}$$

By means of this equation, it is possible to compute every output sample from the previously computed ones and from the initial conditions. The recursion can be performed both along the rows and the columns of the array.

In this case the 2D digital filter is described in the (z_1, z_2) plane by the transfer function

$$H(z_1,z_2) = \frac{\sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-1} a(k_1,k_2) z_1^{-k_1} z_2^{-k_2}}{\sum_{k_1=0}^{M_1-1}\sum_{k_2=0}^{M_2-1} b(k_1,k_2) z_1^{-k_1} z_2^{-k_2}} = \frac{A(z_1,z_2)}{B(z_1,z_2)} \tag{5}$$

which is the ratio of two bivariate polynomials. The matrix $\{b(n_1, n_2)\}$, which defines the recursive part of the filter, is different from zero only in the region $n_1 \geq 0$ and $n_2 \geq 0$. It is normally called a first quadrant sequence and can be indicated with the symbol $\{^{++}b(n_1, n_2)\}$. In the same way it is possible to define filters whose recursive part corresponds to matrices different from zero on the second, third, and fourth quadrants.

The quadrant filters are not the only form which allows the choice of sets R_1 and S_1 and of initial conditions to obtain recursive implementations of Eq. (1) (Mersereau and Dudgeon, 1975). Another choice corresponds to the unsymmetrical half-plane filters, whose coefficient matrices are defined on half-planes.

B. *Two-Dimensional Digital Filter Stability*

Among the different definitions of stability, the most commonly used is based on the BIBO (bounded-input bounded-output) criterion (Cappellini *et al.*, 1978). This corresponds to saying that a filter is stable if its response to a limited input is also limited. It is possible to show that, for causal linear shift-invariant filters, this corresponds to the condition

$$\sum_{n_1=0}\sum_{n_2=0} |h(n_1, n_2)| < \infty \tag{6}$$

where $\{h(n_1, n_2)\}$ is the impulse response of the filter.

The above definition allows the first very important observation that the stability criterion is always verified, if the number of terms in the impulse response is finite, as it is the case with FIR digital filters.

Obviously, the condition of Eq. (6) does not provide a viable method to test the stability of IIR (infinite impulse response) digital filters. In the one-dimensional (1D) case, it is possible to relate the BIBO stability condition to the positions of the singularities of the z transfer function (poles), and it is possible to test the stability by finding the zeros of the denominator polynomial. A similar theorem, which establishes a relation between the stability of the filter and the zeros of the denominator bivariate polynomial, can be formulated also in the 2D case. For causal quadrant filters, this theorem (Shanks *et al.*, 1972) states that, if $B(z_1, z_2)$ in Eq. (5) is a polynomial in z_1 and z_2, the expansion of $1/B(z_1, z_2)$ in negative powers of z_1 and z_2 converges absolutely if and only if

$$B(z_1, z_2) \neq 0, \qquad \text{for } |z_1| \geq 1, |z_2| \geq 1 \tag{7}$$

The corresponding sequences are defined as minimum-phase causal quadrant sequences. Correspondingly, maximum-phase and mixed-phase noncausal quadrant sequences are defined.

Unfortunately, in the 2D case the formulation of the stability conditions as above does not directly produce an efficient stability test, as in the 1D case, due to the lack of an appropriate factorization theorem of algebra.

An approach to the solution of the stability problem in the 2D case can be the use of the properties of the complex cepstrum (Oppenheim and Shafer, 1975) of causal minimum phase sequences and noncausal sequences. The complex cepstrum of a sequence $\{f(n_1, n_2)\}$ is defined as

$$\hat{f}(n_1, n_2) = Z^{-1}[\ln[Z(f(n_1, n_2))]] \tag{8}$$

and it exists if: (1) the Fourier transform of the sequence is not equal to zero or infinity at any frequency; (2) any linear phase component has been eliminated through an appropriate shift of the original sequence (Dudgeon, 1975). These conditions are, for example, satisfied if the coefficient matrix is the coefficient matrix of a squared-magnitude transfer function. Further, if the computed cepstrum of a quadrant causal filter is quadrant causal, then the corresponding sequence is minimum phase and the filter is stable. The same considerations can also be extended to the half-plane filters.

C. *Design Methods of Two-Dimensional Digital Filters*

Many design methods have been defined for 2D digital filters of the FIR and IIR type. Some more important methods are presented in the following, with special reference to those combining a good efficiency with a reasonable complexity of designing and implementation for image processing.

1. *Design of Two-Dimensional FIR Digital Filters*

An important property of 2D FIR digital filters is that they can be designed to have completely real or completely imaginary frequency responses, modified by a linear phase term, if suitable symmetries are present in the impulse response.

As an example, by considering a 2D linear digital filter having N_1 and N_2 odd in its noncausal form, the frequency response of this filter can be written in the form [starting from Eq. (3), with $z_1 = e^{-j\omega_1 X}$ and $z_2 = e^{-j\omega_2 X}$, assuming the space-sampling interval $X = 1$]

$$H(e^{j\omega_1}, e^{j\omega_2}) = \sum_{k_1=0}^{(N_1-1)/2} \sum_{k_2=0}^{(N_2-1)/2} a(k_1,k_2)\cos(k_1\omega_1)\cos(k_2\omega_2) \qquad (9)$$

if the filter impulse response $\{h(n_1,n_2)\}$ has the following symmetries

$$h(n_1,n_2) = h(n_1,-n_2) = h(-n_1,-n_2) = h(-n_1,n_2) \qquad (10)$$

which correspond to symmetries with respect to the origin of the axes and also with respect to the axes (circularly symmetric filters can be obtained in this way).

The frequency response of Eq. (9) is symmetric with respect to the axes, as can be verified with a sign change of ω_1 and ω_2. The frequency response of the causal filter is then obtained by multiplying Eq. (9) by the linear phase term which corresponds to the shift of the impulse response, that is,

$$\exp\{-j([(N_1-1)/2]\omega_1 + [(N_2-1)/2]\omega_2)\}.$$

The design problem consists in the evaluation of the coefficient matrix $\{a(n_1,n_2)\}$ in such a way to meet a set of given specifications in the space or frequency domain. Several different methods have been proposed and defined, some of which are a direct generalization of their 1D counterparts.

A relatively simple design method is the so-called window method (Cappellini *et al.*, 1978). It is based on the consideration that, the 2D frequency response being periodic, it is possible to represent it as a Fourier series, whose coefficients, according to the 2D sampling theorem, are proportional to the samples of the impulse response of the filter. Therefore, it is possible to obtain, analytically or using an approximation method based on the inverse discrete Fourier transform (IDFT), the sampled impulse response, starting from the frequency domain specifications. The problem is that the resulting impulse response is, in general, of infinite order and has to be truncated to obtain a practically usable digital filter. However, if the truncation is performed using a rectangular or circular window function, with an abrupt transition between the value equal to one in the zone where the impulse response has to be retained and equal to zero in the truncation region, quite a large error in the frequency response is obtained. Therefore, the goal is to be able to truncate the impulse response, introducing the minimum error in the frequency response. To this purpose, the obtained values of the sampled impulse response $h(n_1,n_2)$ are multiplied by the samples $w(n_1,n_2)$ of a window function, whose Fourier transform presents a suitable trade-off between the width of the main lobe and the area under the side lobes; that is

$$a(n_1,n_2) = h(n_1,n_2)w(n_1,n_2) \qquad (11)$$

Many window functions have been defined to design filters in the 1D case. For 2D design, extensions to the 2D domain are in general used. In particular, a 2D window, having circular symmetry properties, can be defined starting from a $w(t)$ window as

$$w(x,y) = w(\sqrt{x^2+y^2}) \qquad (12)$$

Three useful window functions are: the Lanczos-extension window (Cappellini window 1), the Kaiser window, and the Weber-type approximation window (Cappellini window 2).

The Lanczos-extension window $w_1(t)$ has the 1D continuous form

$$w_1(t) = \begin{cases} \left[\dfrac{\sin[(\pi t)/\tau]}{\pi t/\tau}\right]^m, & \text{for } |t| \le \tau \\ 0 & \text{for } |t| > \tau \end{cases} \qquad (13)$$

where m is a positive parameter, controlling the correction performance that is the tradeoff between the obtained width of the transition band and the maximum error in the approximation, and τ is half of the window time extension.

The Kaiser window has the form (Kaiser, 1966)

$$w_K(t) = \frac{I_0(\omega_a\tau\sqrt{1-(t/\tau)^2})}{I_0(\omega_a\tau)} \qquad (14)$$

where I_0 is the modified Bessel function of the first kind and zero order and ω_a is a positive number, which controls the tradeoff between the width of the transition bandwidth and the maximum in-band error.

The Weber-type approximation window is a close representation of a function which gives a minimum value of the uncertainty product in a modified form (Hilberg and Rothe, 1971). The obtained expression of the window $w_2(t)$, as a third-order polynomial approximation, is the following (defined in the time interval 0–1.5)

$$w_2(t) = at^3 + bt^2 + ct + d \qquad (15)$$

$0 \le t < 0.75$	$0.75 \le t \le 1.5$
a = 1.783724	a = −0.041165
b = −3.604044	b = 1.502131
c = 0.076450	c = −4.591678
d = 2.243434	d = 3.651582

The windows $w_K(t)$ and $w_2(t)$ represent indeed near-optimum windows, giving high-efficiency digital filters; however, the simple window $w_1(t)$ also gives good-efficiency filters.

As a typical example, Fig. 1 shows the spatial-frequency response (one quadrant) of a circular 2D low-pass digital filter, using the window $w_1(t)$ with $m = 1.6$, for $N_1 = N_2 = 16$ and $\omega_s/\omega_c = 4$ (ω_s = sampling angular spatial frequency; ω_c = cutoff angular spatial frequency).

It is possible to design 2D FIR optimal digital filters by using the linear programming approach (Hu and Rabiner, 1972) and some modifications of the ascent algorithm (multiple exchange ascent algorithm) (Harris and Mersereau, 1977). The main problem with these methods is the computation time. This practically limits the maximum length of the impulse responses of the obtained filters to about 9 × 9 in the linear programming case and to about 15 × 15 in the multiple exchange ascent case, which is more efficient.

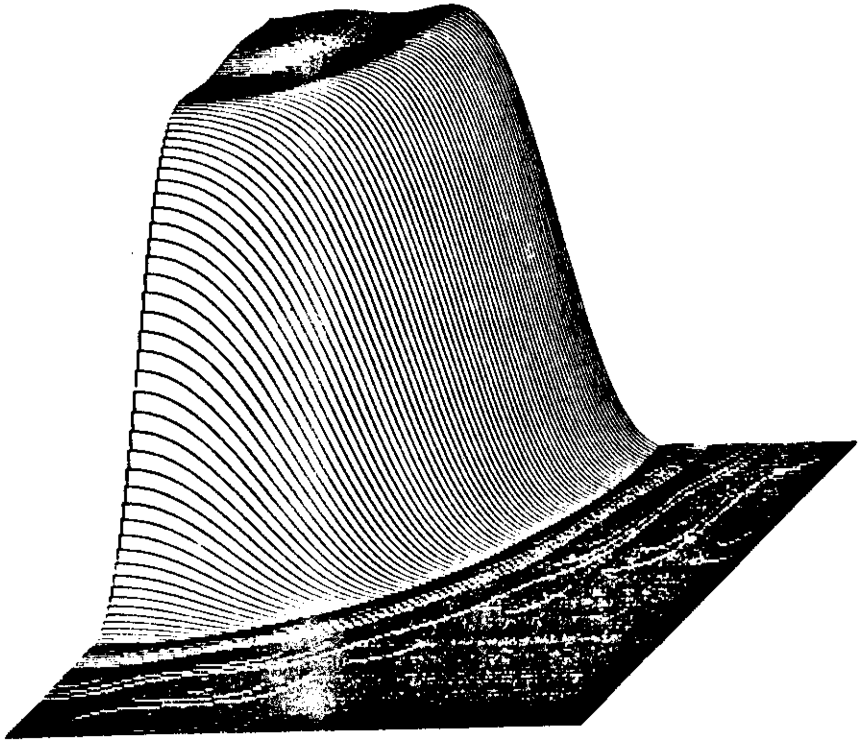


FIG. 1. Spatial-frequency response (one quadrant) of a circular 2D low-pass digital filter, using the window $w_1(t)$.

The design problem can be made more tractable by reducing the number of variables in the linear programming through the frequency-sampling approach (Hu and Rabiner, 1972). In this case a grid of points in the frequency domain is chosen, and most of the frequency sample values are fixed through a direct translation of the filter specifications. A linear programming problem can be set up using constraint relations for the interpolated frequency response, where the variables are the frequency samples in the transition bands. As a typical example, Fig. 2 shows the spatial-frequency response (one quadrant) of a 2D digital filter, having near-circular symmetry, designed through this last procedure for $N_1 = N_2 = 16$ (Calzini *et al.*, 1975).

Another suboptimum design method is based on the transformation of the frequency response of a 1D filter into the frequency response of a 2D filter (McClellan, 1973). Let us consider, for instance, a linear-phase 1D digital filter with N odd: Its frequency response, dropping the linear phase term, can be

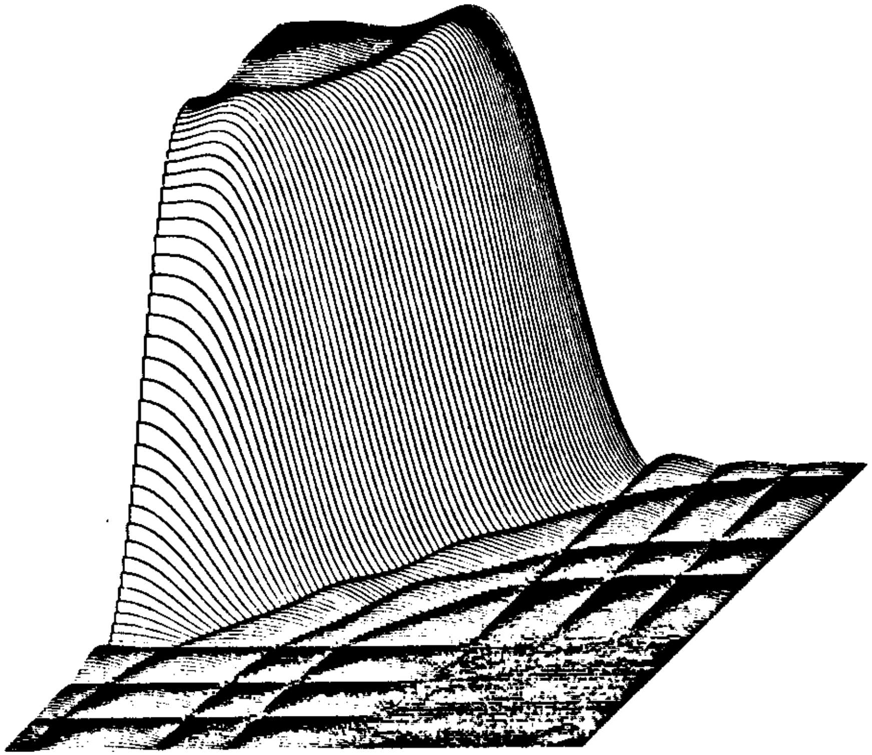


FIG. 2. Spatial-frequency response (one quadrant) of a 2D digital filter, designed by the frequency sampling procedure.

written in the form

$$H(e^{j\omega}) = \sum_{k=0}^{(N-1)/2} a(k)\cos(k\omega) \tag{16}$$

where $a(k)$ are the coefficients defining the frequency response. If a transformation of variables of the form

$$\cos\omega = A\cos\omega_1 + B\cos\omega_2 + C\cos\omega_1\cos\omega_2 + D \tag{17}$$

is carried out in Eq. (16), using the properties of the Chebychev polynomials and of the trigonometric functions, it is possible to obtain a 2D function of the type

$$H(e^{j\omega_1}, e^{j\omega_2}) = \sum_{k_1=0}^{(N-1)/2} \sum_{k_2=0}^{(N-1)/2} a(k_1, k_2)\cos(k_1\omega_1)\cos(k_2\omega_2) \tag{18}$$

which is formally identical to the frequency response of a linear-phase 2D FIR digital filter [see Eq. (9)]. With the choice $A = B = C = -D = \frac{1}{2}$, the mapping contours in 2D are approximately circular, at least for small values of ω_1 and ω_2. This design procedure can also be generalized to the use of transformation relations more complex than the simple relation of Eq. (17) (Mersereau *et al.*, 1976), and some efficient implementation structures exist for the obtained filters (Mecklenbrauker and Mersereau, 1976).

2. *Design of Two-Dimensional IIR Digital Filters*

In this case the coefficients $\{a(n_1, n_2)\}$ and $\{b(n_1, n_2)\}$ have to be chosen to approximate the desired frequency response with a stable recursive implementation [see Eqs. (4) and (5)]. The stability is indeed a specific and important problem of recursive structures, as already shown in Section II,B.

To design 2D digital filters of the IIR type is a more difficult task than to design IIR 1D filters. In fact, the 1D techniques normally rely on the factorability of one-variable polynomials, which result in very simple algorithms for the stability test and for the stabilization of unstable filters; these techniques are unfortunately not directly generalizable to the 2D case (Cappellini *et al.*, 1978).

Two main classes of design methods have been defined. The first one is based on spectral transformations from 1D to 2D and the second one on parameter optimization, using some filter structures, as the second-order filter section cascade, where stability control is easily introduced into the approximation algorithm (Maria and Fahmy, 1974).

A general design procedure (Ekstrom, 1980) uses a nonlinear optimization to minimize an error expression, where the distance from an ideal frequency response and the distance from a stable implementation, obtained by means of the cepstrum decomposition, are present. In this case, it is possible to obtain simultaneous control of the frequency domain approximation and of the stability of the filter, with a procedure which is indeed general but rather complex in implementation; further, some knowledge of the general nonlinear approximation problems, when an acceptable error minimum is not automatically reached, is required.

A proposed design technique (Shanks *et al.*, 1972) consists of mapping 1D filters into 2D filters, with a rotation operation. If a 1D continuous filter is given in its factored form, its transfer function can be viewed as that of a 2D filter that varies in one direction only

$$H(s_1, s_2) = H_1(s_2) = H_0\left(\prod_{i=1}^{m}(s_2 - q_i) \Big/ \prod_{i=1}^{n}(s_2 - p_i)\right) \tag{19}$$

where s_1, s_2 are the Laplace variables and q_i, p_i are the zeros and poles, respectively. A rotation of the (s_1, s_2) axes through an angle β can be performed by means of the transformations

$$\begin{aligned} s_1 &= s'_1\cos\beta + s'_2\sin\beta \\ s_2 &= -s'_1\sin\beta + s'_2\cos\beta \end{aligned} \tag{20}$$

A filter whose frequency response is now a function of s_1 and s_2 and corresponds to a rotation by an angle $-\beta$ of Eq. (19) is obtained. Then a digital filter can be defined through the application of the bilinear z transform to both the continuous variables.

The above approach, which in direct implementation suffers from the warping effects of the bilinear z transform, has been used to obtain simple rotated blocks, which can be combined to define circularly symmetric recursive filters (Costa and Venetsanopoulos, 1974), where also the conditions for the stability of the rotated sections have been proved.

Another method (Bernabò *et al.*, 1976) is based on the transformation of the squared magnitude function of a 1D digital filter to the 2D domain, followed by a suitable decomposition of the resulting filter. With reference to Section II,A, given a first-quadrant filter (causal filter), it is possible to define the corresponding second-, third-, and fourth-quadrant filters, according to the relations

$$h_1(n_1, n_2) = h_2(n_1, -n_2) = h_3(-n_1, -n_2) = h_4(-n_1, n_2) \tag{21}$$

with transfer functions

$$H_1(z_1, z_2) = H_2(z_1, z_2^{-1}) = H_3(z_1^{-1}, z_2^{-1}) = H_4(z_1^{-1}, z_2) \tag{22}$$

The cascade of the four filters is a zero-phase digital filter, whose frequency response is defined by the coefficients $p(k_1, k_2)$ and $q(k_1, k_2)$, determined

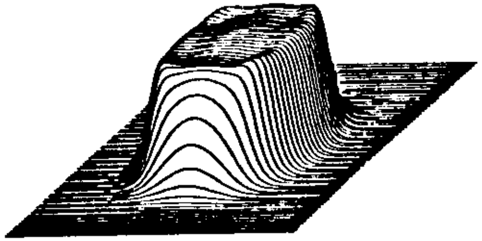


FIG. 3. Spatial-frequency response of a 2D IIR digital filter.

through the convolution of the coefficients of the four filters, and has the following form [(see also Eq. (5)]

$$G(e^{j\omega_1}, e^{j\omega_2}) = \frac{\sum_{k_1=0}^{N-1} \sum_{k_2=0}^{N-1} p(k_1, k_2) \cos(k_1\omega_1) \cos(k_2\omega_2)}{\sum_{k_1=0}^{M-1} \sum_{k_2=0}^{M-1} q(k_1, k_2) \cos(k_1\omega_1) \cos(k_2\omega_2)} \tag{23}$$

Such a 2D frequency response can be obtained through the transformation of Eq. (17) applied to the numerator and denominator of the squared magnitude function of a 1D IIR digital filter. The obtained squared magnitude transfer function has to be factorized to get stable recursive digital filters and the cepstrum decomposition [see Eq. (8)] can be used. In particular, to reduce the error connected to the truncation of the infinite cepstrum, windows (see Section II,C,1) can be used to reduce oscillations.

Figure 3 shows an example of a 2D IIR digital filter, designed according to the above procedure: The squared magnitude function of a fourth-order Chebychev low-pass filter, having a 2% in-band ripple, a normalized cutoff space frequency 0.25, and a −20-dB space frequency 0.35, is used; the filter has a numerator and a denominator with 6 × 6 coefficients, obtained using a Kaiser window [Eq. (14)] with $\omega_a = 3$. The filter maximum in-band ripple turns out to be 0.022 and the transition band, defined as the difference between the normalized space frequencies, where the amplitude of the frequency response is, respectively, 90% and 10% of the in-band nominal value, is equal to 0.0937.

III. LOCAL SPACE OPERATORS

Local space operators can be considered as simpler 2D digital filters defined only in the space domain (see Section I), while the above described are more complex 2D digitals filters defined and designed both in the space and frequency domains.

Indeed local space operators have in general low complexity: small blocks of data (image samples) are processed in the space domain. The interest of this approach is connected to the resulting economic implementation and very fast processing capabilities, characteristics which are very important when large amount of data (image samples) have to be processed or when real-time operators are to be performed. Further local space operators of the nonlinear type can be defined, solving difficult image processing problems (as noise reduction) in a fast and efficient way.

Some very simple local space operators are that ones used to perform sampling or quantization variations. Through sample repetition or interpolation a zooming effect can be obtained in order to change the sampling rate or to present particulars of the observed scenes. Through quantization variation, scale expansion or compression can be obtained to perform image enhancement or smoothing, respectively. In particular a useful image enhancement can be accomplished, by evaluating the amplitude distribution or histogram on the entire analyzed image and changing the scale in such a way to shift the minimum amplitude to zero and the maximum amplitude to the full scale value (image stretching).

The other local space operators can be classified into two main groups: operators performing image smoothing or enhancement (low-pass or high-pass filtering) and operators performing edge extraction (derivative filtering).

A. *Local Space Operators for Image Smoothing and Enhancement*

One simple local operator of this type is the average operator, which can be used to reduce variations in the grey levels, for instance reducing noise components in the image. As an example, the arithmetic average of 9 samples (pixel values) can be performed: The obtained result is then substituted for the central pixel of the 3 × 3 sample matrix used according to the relation

$$g(n_1, n_2) = \frac{1}{9} \sum_{k_1=-1}^{1} \sum_{k_2=-1}^{1} f(n_1 - k_1, n_2 - k_2) \tag{24}$$

where $f(n_1, n_2)$ are the image samples. The operation is then iterated for all the points of the image, correponding obviously to a sliding convolution (FIR filtering). Indeed, operators of this type are equivalent to low-pass filters, reducing variations between adjacent pixels of the image. Their importance, as already observed, lies in the fact that very fast implementations result, because no multiplications need to be performed.

On the other hand, another class of operators exists to emphasize the differences between adjacent pixels. This is obviously the opposite of the previously considered operation, and its purpose is to enhance the variations, outlining the contours of the objects in the image.

Along this line, very simple separable algorithms can be defined on a row and/or column basis. As an example, let us consider the following simple procedure: the differences d between the corresponding pixels (same column) of two adjacent rows are computed; then, if $d > 0$ the maximum luminance value (white level) is substituted for the sample; if $d < 0$ the black value is chosen as the sample value; and if $d = 0$ an intermediate value is selected. Obviously, this procedure enhances the transitions between the rows, that is along the columns of the image. However, the procedure can be repeated along the rows, considering the differences between the columns.

Another approach is based on the computation of the differences between a pixel and the average value of the 8 adjacent values according to the relation

$$g(n_1, n_2) = f(n_1, n_2) - \frac{1}{8} \sum_{\substack{k_1 = -1 \\ k_1 + k_2 \neq 0}}^{1} \sum_{k_2 = -1}^{1} f(n_1 - k_1, n_2 - k_2) \qquad (25)$$

The above two operators correspond to high-pass filtering algorithms, of which the first is separable.

Of increasing interest are local space operators performing nonlinear filtering (Cappellini, 1983). One interesting example is represented by the following nonlinear smoother of noisy images, which is especially useful before edge detection. By considering a block of 3 × 3 samples, the smoother is defined by the relation

$$g(n_1, n_2) = \frac{1}{n} \sum_{f \in S} f(n_1 - k_1, n_2 - k_2) \qquad (26)$$

where $S = \{f : |f(n_1 - k_1, n_2 - k_2) - f(n_1, n_2)| < c_1\}$ and $k_1, k_2 = -1, 0, 1$ with $k_1 + k_2 \neq 0$. By means of this smoother, the value of each pixel is replaced by the average of its neighborhood values, except those which have level differences greater than a fixed value (c_1) in absolute value. In this way, small-amplitude noise is removed, while no degradation results for edges and boundaries present in the processed image regions (a smoothing of the pixel values on either side of the edge is performed without any damage for the edge itself, as would happen with linear smoothing). Further, this nonlinear filtering procedure can be iterated to obtain a greater noise reduction; two iterations are in general sufficient.

Another interesting example of nonlinear operators is represented by the following nonlinear filters, very useful in reducing noise spikes or scintillation pulses essentially represented by high noise levels concentrated in one or two pixels (due to image sensors such as TV cameras, photodetectors, ir detectors, ultrasonic transducers, or to the analog-to-digital conversion).

Let us evaluate the average value $f_a(n_1, n_2)$ of three grey levels of the pixels in a 3 × 3 block [excluding the central $f(n_1, n_2)$] as expressed by the double addition in Eq. (25): if all 8 pixels, around $f(n_1, n_2)$, have a grey level differing from $f_a(n_1, n_2)$ less than a suitable threshold T and $f(n_1, n_2)$ differs from $f_a(n_1, n_2)$ more than $T + \Delta$ (with $\Delta > 0$), the value of $f(n_1, n_2)$ is set equal to the $f_a(n_1, n_2)$; otherwise the central pixel maintains its original value average $f_a(n_1, n_2)$; otherwise the central pixel maintains its original value $f(n_1, n_2)$. By adjusting the two parameters T and Δ, noise spikes of a different level in more or less flat image regions can be eliminated.

For a binary image, as obtained after edge detection, a nonlinear operator analogous to the preceding one can be defined through the following relations (Cappellini, 1983)

$$\text{if } f(n_1, n_2) = 0 \begin{cases} f'(n_1, n_2) = 0, & \text{if } \displaystyle\sum_{\substack{k_1 = -1 \\ k_1 + k_2 \neq 0}}^{1} \sum_{k_2 = -1}^{1} f(n_1 - k_1, n_2 - k_2) < n_m \\ f'(n_1, n_2) = 1, & \text{otherwise} \end{cases} \qquad (27)$$

$$\text{if } f(n_1, n_2) = 1 \begin{cases} f'(n_1, n_2) = 0, & \text{if } \displaystyle\sum_{\substack{k_1 = -1 \\ k_1 + k_2 \neq 0}}^{1} \sum_{k_2 = -1}^{1} f(n_1 - k_1, n_2 - k_2) < n_0 \\ f'(n_1, n_2) = 1, & \text{otherwise} \end{cases} \qquad (28)$$

where $f'(n_1, n_2)$ is the updated value regarding the central pixel $f(n_1, n_2)$ and in general $n_m = 7$ and $n_0 = 2$ or $n_m = 8$ and $n_0 = 1$ (the last case means that a central pixel of value 0 is changed to 1 if all the 8 near pixels are 1 and a central pixel of value 1 is changed to 0 if all the 8 near pixels are 0).

B. Edge Detectors

A very important class of 2D local space operators is represented by edge detectors, which extract edges or boundaries in the processed image. Most of these operators perform a kind of derivative filtering, evaluating the gradient through a test on a given image pixel and its close ones. Once the gradient has been estimated (magnitude and angle), it is compared with a threshold: If its value (in particular the magnitude) is greater than the threshold, the pixel is considered as a part of an edge, whose direction is orthogonal to the gradient direction. These operators can be divided in two groups: To the first group belong the operators which evaluate the two orthogonal components of the gradient; the second group is based on gradient detection by means of a set of templates or masks of different orientation (Pratt, 1978; Cappellini, 1979b).

In the first group, two orthogonal components D_x and D_y of the gradient in each pixel are evaluated, and then its magnitude is obtained by means of the relation

$$D = \sqrt{D_x^2 + D_y^2} \tag{29}$$

while its direction is evaluated as

$$Y = \arctan(D_y/D_x) \tag{30}$$

The two components D_x and D_y can be evaluated by several methods, by using different weights on a given number of values near the tested pixel. By considering the (n_1, n_2) pixel, the simplest way to obtain the two components D_x and D_y corresponds to evaluating the differences between the adjacent pixels, that is,

$$\begin{aligned} D_x &= f(n_1, n_2 + 1) - f(n_1, n_2) \\ D_y &= f(n_1, n_2) - f(n_1 + 1, n_2) \end{aligned} \tag{31}$$

This is the same as using the coefficient matrices or masks

$$D_x = \begin{bmatrix} -1 & 1 \\ 0 & 0 \end{bmatrix} \qquad D_y = \begin{bmatrix} 1 & 0 \\ -1 & 0 \end{bmatrix} \tag{32}$$

in a nonrecursive digital filter implementation (performing the addition of the products of the values of the masks and the underlying values of the image pixels).

Another method to find the gradient components D_x and D_y is the following (Roberts method)

$$\begin{aligned} D_1 &= f(n_1, n_2 + 1) - f(n_1 + 1, n_2) \\ D_2 &= f(n_1, n_2) - f(n_1 + 1, n_2 + 1) \end{aligned} \tag{33}$$

with the corresponding masks

$$D_1 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \qquad D_2 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{34}$$

which gives two orthogonal components rotated $\pi/4$ with respect to the image axes.

A more accurate estimation of the gradient can be obtained by using 3 × 3 coefficient matrices. Some of these matrices or masks are reported in the following

(1) smoothed gradient

$$D_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, \qquad D_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \tag{35}$$

(2) Sobel gradient

$$D_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \qquad D_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{36}$$

(3) isotropic gradient

$$D_x = \begin{bmatrix} -1 & 0 & 1 \\ -\sqrt{2} & 0 & \sqrt{2} \\ -1 & 0 & 1 \end{bmatrix}, \qquad D_y = \begin{bmatrix} 1 & \sqrt{2} & 1 \\ 0 & 0 & 0 \\ -1 & -\sqrt{2} & -1 \end{bmatrix} \tag{37}$$

The expressions related to the previous masks are of the form

$$\begin{aligned} D_x = {} & f(n_1 - 1, n_2 + 1) + wf(n_1, n_2 + 1) + f(n_1 + 1, n_2 + 1) \\ & - f(n_1 - 1, n_2 - 1) - wf(n_1, n_2 - 1) - f(n_1 + 1, n_2 - 1) \\ D_y = {} & f(n_1 - 1, n_2 - 1) + wf(n_1 - 1, n_2) + f(n_1 - 1, n_2 + 1) \\ & - f(n_1 + 1, n_2 - 1) - wf(n_1 + 1, n_2) - f(n_1 + 1, n_2 + 1) \end{aligned} \tag{38}$$

where the weight w assumes the values 1, 2, $\sqrt{2}$ for the three masks, respectively.

The second group of operators for the estimation of the gradient and then for the identification of edges is based on gradient detection by means of a set of templates or masks of different orientation, searching sequentially at each pixel for the best match among the image subarea and the masks. Every mask of the set is superimposed on each pixel of the image, and the additions of products between the mask coefficients and the underlying pixels of the image are performed just as in the previous group of local operators. The gradient is assumed to be detected by the mask which gives the greatest value of the addition of the products: Its direction is assumed according to the direction of the mask. Each set of masks is composed of eight different 3 × 3 masks, each of which is obtained from the previous one through a circular permutation of its elements around the central one. Thus, if we assume that the first mask of a given set is

$$\begin{bmatrix} \mathrm{A} & \mathrm{B} & \mathrm{C} \\ \mathrm{D} & \mathrm{E} & \mathrm{F} \\ \mathrm{G} & \mathrm{H} & \mathrm{I} \end{bmatrix} \tag{39}$$

the second and the third mask will be

$$\begin{bmatrix} \mathrm{B} & \mathrm{C} & \mathrm{F} \\ \mathrm{A} & \mathrm{E} & \mathrm{I} \\ \mathrm{D} & \mathrm{G} & \mathrm{H} \end{bmatrix} \qquad \begin{bmatrix} \mathrm{C} & \mathrm{F} & \mathrm{I} \\ \mathrm{B} & \mathrm{E} & \mathrm{H} \\ \mathrm{A} & \mathrm{D} & \mathrm{G} \end{bmatrix} \tag{40}$$

and so on.

The sets of masks more frequently used are obtained through a permutation of the following masks (Cappellini, 1979b)

(1) Prewitt mask

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{bmatrix} \tag{41}$$

(2) Kirsch mask

$$\begin{bmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} \tag{42}$$

(3) Robinson masks

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{43}$$

There are also other approaches to edge detection, in particular to increase the processing speed or to process noise images. An interesting example is represented by the following operator (Cappellini and Odorico, 1981). A block of 3 × 3 pixels is considered: To each of the 8 pixels surrounding the central one a binary value is given according to the difference among the pixel value and the central value. In this way a binary image is obtained having 256 possible configurations (taking the central pixel at a constant reference value of 0 or 1): these configurations are divided into 5 classes, having a decreasing probability that the central pixel is a part of an edge or a contour. By setting a threshold, separating these classes into two groups, it is finally estimated whether the central pixel is or not a part of an edge. Interesting aspects of this operator are the following ones: adaptive criteria can be used for the above separation into two groups, depending on the noise characteristics in the processed image; high-speed implementation is obtained, due to the fact that to estimate whether a pixel is or not a part of an edge, after the binary values were obtained, it is practically sufficient to compare the actual binary configuration with a memorized decision table.

IV. Data Compression

Data compression is a digital operation or transformation performed to reduce the amount of redundant data (redundancy reduction) and is particularly useful for image processing, due to the large number of sampled

data representing each image. In the following some general considerations on source coding and several data compression methods and techniques are presented, outlining the more important for image processing.

A. *Source Coding*

Let us consider an information source S emitting the information symbols $s_1, s_2, \ldots, s_q$. These discrete symbols could be, for instance, the different grey levels of a sampled image. Let us assume that each symbol is emitted with a specific probability: $p(s_i)$ is the probability of s_i (in practice, as is well known, this probability could be estimated as the ratio between the number of times that s_i appears and the overall number of observed or recorded symbols).

The information quantity given by the symbol s_i is assumed to be proportional to the inverse of its probability $p(s_i)$ through a logarithmic function; that is,

$$I(s_i) = k_0 \log_a \frac{1}{p(s_i)} \tag{44}$$

where k_0 is a constant and a is the logarithmic base. In general the following values are assumed: $k_0 = 1$, $a = 2$. In this way, the information quantity connected to a symbol having probability $\frac{1}{2}$ (equal probability in the case of only two symbols) results to be unitary value, that is 1 bit (binary unit).

Once having defined the information quantity of a symbol s_i, it is possible to define the mean information quantity or entropy of a source $S = \{s_1, s_2, \ldots, s_q\}$ of the above type, also called zero-memory source due to the fact that the emission of one symbol is independent by the emission of the other near symbols. The entropy of the zero-memory source, $H(S)$, is defined as (Shannon and Weather, 1949)

$$H(S) = \sum_{i=1}^{q} p(s_i) \log_2 \frac{1}{p(s_i)} \tag{45}$$

which measures the average number of bits per symbol. It is easy to prove that the entropy function $H(S)$ has the maximum value when the symbols s_i are equiprobable, that is when $p(s_i) = 1/q$; in this case $H(S) = \log_2 q$.

One zero-memory source of particular interest is represented by the binary source, having only two symbols $s_1 = 0$ and $s_2 = 1$. Denoting $p = p(0)$, it results that $p(1) = \bar{p} = 1 - p$. The entropy function $H_2(S)$ has the expression

$$H_2(S) = p \log_2 \frac{1}{p} + \bar{p} \log_2 \frac{1}{\bar{p}} \tag{46}$$

This function, also denoted $H(p)$, is often called the entropy function and has a maximum value $H(p) = 1$ for $p = \bar{p} = \frac{1}{2}$.

A slightly more complex information source is represented by the nth extension of the zero-memory source, in which blocks of n symbols s_i are considered (as the words representing the image samples). The symbols σ_i of this extended source S^n are q^n and the entropy is

$$H(S^n) = \sum_{i=1}^{q} p(\sigma_i)\log_2 \frac{1}{p(\sigma_i)} = nH(S) \qquad (47)$$

A more general and complex information source is represented by the Markov source or source with memory, in which the emission of any symbol s_i is done with a conditional probability $p(s_i/s_{j_1}, s_{j_2}, \ldots, s_{jm})$; that is, this emission is now connected to a specific group of preceding symbols $s_{j_1}, s_{j_2}, \ldots, s_{jm}$. The Markov source is called of m order (m denotes the memory extension). The above conditional probabilities are q^{m+1}, because there are q^m possible configurations or states of the source, corresponding to the different possible dispositions with repetition of the m symbols, and from each one of this state can appear any one of the q source symbols. The representation of the Markov sources can be done through the state diagrams, in which the different source states are reported as points and the possible connections among the states are outlined by conjunction lines (Abramson, 1963).

Let us now consider the source coding. By considering, for instance, a zero-memory source $S = \{s_1, s_2, \ldots, s_q\}$ with symbol probabilities $p(s_1)$, $p(s_2), \ldots,$ $p(s_q)$ as above, each symbol s_i can be transformed or mapped into a fixed sequence of l_i symbols taken from a finite alphabet $X = \{x_1, x_2, \ldots, x_r\}$. This corresponds to encoding each symbol s_i into a code word X_i, belonging to the set $\{X_1, X_2, \ldots, X_q\}$; X is called the code alphabet.

Source codes can be classified according to the code-word structure: codes using a variable-length encoding, in which the code words X_i have a variable length; codes with a fixed length of the code words. Further, the source codes can be distinguished according to the following properties:

(1) nonsingular codes, having all different code words;

(2) codes which can be univocally decoded, for which the nth code extension is nonsingular for any finite value of n;

(3) comma codes, having a specific symbol to separate a code word from the near ones;

(4) instantaneous codes, for which any code word can be decoded into a source symbol without the necessity of considering or knowing the following symbols.

An important example of a source code is represented by the usual binary code, used to represent an image sample (quantized grey level) in a digital form.

This binary code, having in general a constant word length, is a simple example of a nonsingular code, which can be univocally decoded.

A very important property of source codes is connected to the economy or compactness of representation of the information symbols. To make this concept precise, the average code word length can be defined as

$$L = \sum_{i=1}^{q} p(s_i)l_i \qquad (48)$$

This parameter L is very useful to measure the economy of each source code. For instance, a source code having an average word length L less than or equal to that of all other codes using the same code alphabet for the same information source is called a compact code. It is clear now that a fundamental problem of the source coding consists in the search and definition of compact codes for the different information sources.

A general theoretical solution to the above problem is given by the first Information Theory Theorem or first Shannon theorem for source coding (Shannon and Weather, 1949; Shannon, 1959). Substantially this theorem establishes a general bound for the average word length L in relation to the information source entropy. In simplified form this bound is expressed by the following relation

$$H_r(S) \leq L \qquad (49)$$

where $H_r(S)$ is the source entropy, measured with a logarithm in base r. The bound can also be set as

$$H_r(S) \leq L_n/n \qquad (50)$$

where L_n represents the average word length of the nth extension of the information source with the following limit

$$\lim_{n\to\infty}(L_n/n) = H_r(S) \qquad (51)$$

In general the price which is paid to reduce L or L_n/n is represented by the complexity of the source coding.

The above fundamental theorem permits us now to also define in a rigorous way the efficiency of each source code [according to Eq. (49)]

$$\eta = H_r(S)/L \qquad (52)$$

and the redundancy of the source code as

$$1 - \eta = [L - H_r(S)]/L \qquad (53)$$

The above Eqs. (52) and (53) permit one to compare different codes for the same information source, selecting that which has the higher efficiency (higher value of η) or less redundancy.

An example of compact or optimum codes is represented by the Huffman encoding procedure, in which the length l_i of each code word is inversely related to the value of the probability $p(s_i)$. In this way the more probable and therefore the more frequent words are encoded in shorter sequences compared with the less probable ones (Abramson, 1963).

B. *Data Compression Methods and Techniques*

Many methods and techniques of source coding or data compression have been studied, defined, and applied to image processing (for local processing, image transmission or storage).

According to the first Shannon theorem (see Section IV,A), the different data compression methods can be divided into two main classes:

(1) reversible methods, which permit—at least in principle—recovering through decompression (source decoding or inverse transformation) all the original source information amount;
(2) irreversible methods, which do not permit recovering all the original data and which introduce therefore some information loss or distortion.

The reversible methods obey and respect the source coding bound of Eq. (50), while the irreversible methods do not.

Among the reversible methods, some important ones for image processing are:

(1) adaptive sampling and quantization
(2) prediction interpolation (adaptive and nonadaptive)
(3) variable word-length coding (as the Huffman code)
(4) digital filtering (maintaining a useful spectrum extension)
(5) use of transformations (Fourier, Walsh–Hadamard, Haar, Karhunen–Loeve)

while some irreversible methods are:

(1) thresholding
(2) parameter extraction
(3) power spectrum
(4) digital filtering (with spectrum extension reduction)
(5) probability functions.

To evaluate the performance of data compression methods, distortion functions and distortion measures can be used (Berger, 1971). In practice, three measurements are evaluated: the compression ratio C_r; the peak error e_p; the rms error e_r. The compression ratio C_r is defined as (Benelli *et al.*, 1980)

$$C_r = L_S/L \tag{54}$$

where L_S is the mean source word length [in general equal to the entropy $H(S)$] and L is the mean word length data compression. If the source messages (image samples) are represented in the standard binary form, the compression ratio C_r can be obtained by the ratio

$$C_r = N_S/N_C \tag{55}$$

where N_S represented the number of bits (0 and 1 values) of the source words (representing the image samples), and N_C represents the corresponding number of bits of the code words after compression.

The peak error e_p represents the peak or maximum error resulting between the input message (image samples) and the corresponding reconstructed message after decoding or decompression, while the rms error e_r represents the root-mean-square error between the input and reconstructed messages evaluated on a suitable block of data (number of image samples, in particular on the image samples overall).

It is clear that in general data compression methods of the irreversible type give higher values of the compression ratio C_r, with the penalty however of higher e_p and e_r errors. Furthermore, reversible methods can also become irreversible, if some parameters (as threshold values, sampling frequency, amplitude tolerances, etc.) are changed in such a way as to obtain higher compression ratios (consequently introducing higher error values).

In practice, the efficiency of the different data compression methods depends on the *nature of the information source* (different types of images), and each method can result in being more efficient for some information sources (particular type of image) than for other ones. In this regard, it is very important to perform a suitable analysis of the information source (image to be processed) before the application of any data compression method:

(1) space analysis
(2) space-frequency analysis (amplitude and phase spectrum)
(3) statistical analysis (statistical average, amplitude distribution, autocorrelation, power spectral density).

1. *Adaptive Sampling and Quantization*

Adaptive sampling methods are based on the use of the minimum sampling frequency for any processed image; the minimum sampling frequency, as is well known (from the sampling theorem), is equal to double the maximum space frequency. Adaptivity can in particular be obtained by changing the sampling frequency for some sub-block of the processed image; for instance, sample sub-blocks of 32 x 32, 64 x 64, or 128 x 128 are considered, and the sampling frequency is locally adjusted in each image sub-block according to the local maximum space frequencies. The maximum space frequency in the image or image sub-blocks can be practically found through the 2D FFT (fast Fourier transform). By means of the above procedures, a reduction of sample number is obtained, in comparison with the use of a fixed sampling frequency.

Also, the quantization level (number of bits representing the image samples) can be changed in an adaptive way from one image to another or in image sub-blocks, taking into account the actual grey-level range (for instance, low grey-level ranges require a lower number of bits). The grey-level range can be easily found through the amplitude distribution (grey-level histogram).

In practice, the change of sampling frequency or quantization law from one image to another or in image sub-blocks can be identified through a special word expressing the particular sampling frequency or quantization level locally used.

2. *Prediction and Interpolation*

Data compression methods with prediction or interpolation are interesting, due to their relatively simple structure and reasonably good efficiency (Benelli *et al.*, 1980). Let us consider 1D algorithms, which can be applied to image processing line by line (for instance, processing the sequence of image samples along the rows).

In prediction methods *a priori* knowledge of some previous samples (image samples along a row) is used, while in interpolation methods *a priori* knowledge both of previous and future samples is utilized. In both types of operations the most widely applied technique consists in comparing the predicted or interpolated sample with the actual sample: If the difference is less than a fixed error or amplitude tolerance, the actual sample is not maintained, otherwise the actual sample is maintained, Figure 4 shows a block diagram of a typical data compression system with prediction or interpolation: The nonredundant samples (i.e., the samples for which the prediction or interpolation fails) are fed into a buffer to be reorganized at constant space intervals with the space position identification (synchronization), necessary for the reconstruction of the original data from the compressed samples. The important role of the buffer is therefore to store the incoming samples remaining after the gate, so that they can be reorganized at uniform sampling rate. In this way, while at the input of the buffer we have only bit compression, at its output we also have bandwidth compression (sampling rate reduction). In the following, the symbols C_{rb} and C_{rs} indicate the average compression ratios, respectively, with and without including the bits added for the space identification or synchronization.

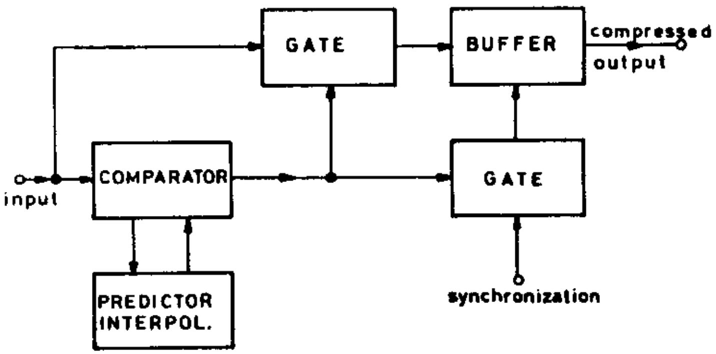


FIG. 4. Block-diagram of a data compression system with prediction or interpolation.

Prediction algorithms are utilized, according to the following difference equation

$$f_p(n) = f(n-1) + \Delta f(n-1) + \Delta^2 f(n-1) + \cdots + \Delta^N f(n-1) \quad (56)$$

where $f_p(n)$ = predicted sample at space position nX (X being the space sampling interval along the image rows and columns); $f(n-1)$ = sample value at the previous space position $(n-1)X$; $\Delta f(n-1) = f(n-1) - f(n-2), \ldots, \Delta^N f(n-1) = \Delta^{N-1} f(n-1) - \Delta^{N-1} f(n-2)$. The value of N corresponds to the order of the prediction algorithm: with $N = 0$ we obtain the zero-order predictor (ZOP) and with $N = 1$ the first-order predictor (FOP).

For the ZOP algorithm, several procedures can be followed:

(1) ZOP with fixed aperture, in which the dynamic range of the data is divided into a set of fixed tolerance bands with a width of 2δ; if $f(n-1)$ is the last remaining sample, $f(n)$ is not maintained when it lies in the same tolerance band;

(2) ZOP with floating aperture, where a tolerance band $\pm\delta$ is placed about the last remaining sample; if the following sample lies in this band, it is not maintained (in this case the next samples are compared again with the value of the last remaining sample $\pm\delta$ and so on);

(3) ZOP with offset aperture, in which the predicted sample is $f_p(n) = f(n-1) \pm \delta$, where δ is a prefixed quantity and the sign + is used if the last remaining sample is out of tolerance in the positive direction and vice versa.

In the FOP algorithm with floating aperture the first two samples are maintained and a straight line is drawn through them, placing an aperture $\pm\delta$

about that line: If the actual sample $f(n)$ is within this aperture, it will not be maintained and the line will be extrapolated for a following space interval X and so on.

Data compression algorithms using interpolation differ from the corresponding ones with prediction, due to the fact that with interpolation both previous and following samples are used to decide whether or not the actual sample is redundant. The more interesting algorithms, based on low-order interpolation, are zero-order interpolator (ZOI) and first-order interpolator (FOI).

Adaptive data compression methods with prediction or interpolation represent an improvement on the preceding ones of the nonadaptive type, especially when there are input images to be processed having high activity (variation of the space and frequency behavior from one image part to the other). These methods can be divided into linear and nonlinear ones depending on the specific procedure used for the adaptivity implementation. Of high interest is the adaptive-linear prediction (ALP) method, in which the predicted sample $f_p(n)$ is evaluated by a linear weighting of M previous samples (Benelli *et al.*, 1980)

$$f_p(n) = \sum_{k=1}^{M} \beta_k f(n-k) \tag{57}$$

where β_k are suitable weighting coefficients. If the prediction error falls within a given threshold value γ, the actual sample is not maintained If the considered process is a stationary Gaussian series with zero mean, the coefficients β_k can be determined in such a way as to minimize the mean-square prediction error given by

$$\delta^2(M,N) = \frac{1}{N}\sum_{i=1}^{N}\left(f(n-i) - \sum_{k=1}^{M}\beta_k(M,N)f(n-i-k)\right)^2 \tag{58}$$

M being the number of the preceding samples for the prediction and N the number of samples which the predictor uses to learn the image sample's evolution (line by line). The method turns out to be advantageous as long as the statistical characteristics of the image are maintained on suitable extended regions. In practice a counter can be used to measure the number of consecutive predictions affected by error; when this number exceeds a prefixed value T, a new set of M coefficients is again computed and the algorithm goes with the new coefficients.

3. *Differential Pulse-Code Modulation and Delta Modulation*

The general block diagram of differential pulse-code modulation (DPCM) is shown in Fig. 5. A predicted sample $f_p(n)$ is evaluated through a linear weighting of the M previous samples (Benelli *et al.*, 1980)

$$f_p(n) = \sum_{k=1}^{M} a(k) f(n-k) \tag{59}$$

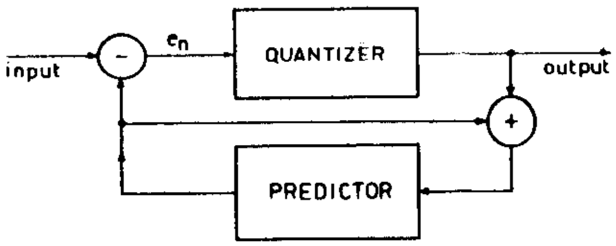


FIG. 5. Block diagram of a DPCM system.

The predicted samples can be obtained using any of the prediction algorithms as ZOP, FOP, ALP, ... The difference e_n between the actual sample and the predicted one is quantized with quantization intervals of amplitude Δ and encoded in a code word L_D bits long. If the image samples have high correlation and the weighting coefficients are correctly chosen, DPCM generally offers a higher efficiency with respect to the usual binary coding; with an equal number of bits, DPCM assures an higher signal-to-quantization-noise ratio (SNR), or with an equal SNR it requires a lower number of bits.

Many adaptive DPCM methods (ADPCM) have also been studied and defined. In general the Δ value is varied, becoming smaller when the grey level is quiescent and vice versa or the length of the prediction interval (M) is changed, according to the signs and values of some previous differences between the predicted and the actual samples.

A special data compression method, which can be considered as a DPCM with 1-digit code, is represented by delta modulation (DM). In the DM method, the changes in the grey level between consecutive samples are substituted for the absolute grey-level values. These changes are represented in the form of binary pulses, whose sign (+ or −) depends on the sign of the amplitude change. Fig. 6a shows the block diagram of a DM system, while Fig. 6b outlines the main wave forms at different points of the system.

In the classical DM a single binary pulse is obtained at each sampling interval instead of a complete code word; the output pulse is in this case synchronous with the input word stream, yielding a constant compression ratio. Errors in the reconstructed data can, however, appear due to two effects: the approximation of the input wave form (grey-level variation) to a step function (granular or quantization noise); and quick variations of the input wave form, which cannot be followed with accuracy. Regarding this last

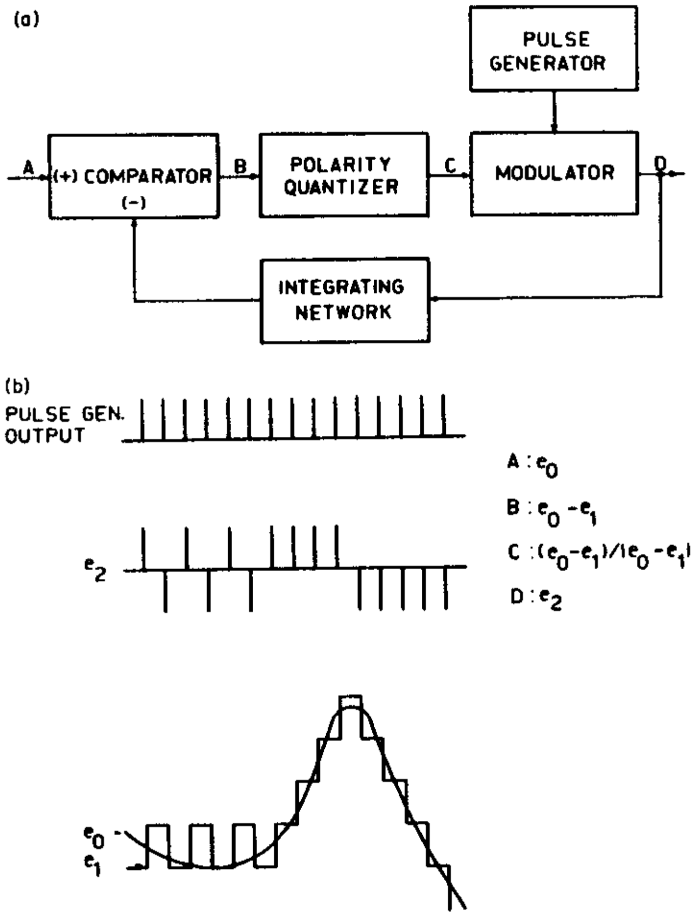


FIG. 6. DM system: (a) block diagram; (b) main wave forms in the different points.

aspect, input variations cannot be followed for which the gradient of the sampled data exceeds the limit

$$g = \Delta r \tag{60}$$

where Δ is the change in amplitude of a DM pulse and r is the rate of the pulses (pulse space frequency for the processed image). The distortion due to this aspect is also known as slope-overload distortion.

Many studies have been developed to analyze the efficiency of the classical DM and to increase the efficiency (Benelli *et al.*, 1980). A first method is based on the change of the step amplitude, according to the wave-form variations: The step amplitude is increased when a given number N_0 of consecutive samples have the same binary value and it is decreased in the contrary case (high information delta modulation, HIDM). Another interesting modification of the classical DM is basic asynchronous delta modulation (BADM), in which the sampling rate is increased during intervals of high activity (rapid dynamic range variations) and it is decreased in lower activity intervals. A special technique, called operational asynchronous delta modulation (OADM), avoids the errors corresponding to rapid amplitude variations in the following way: When the difference between the input and the reconstructed samples exceeds a prefixed tolerance value, the algorithm goes back m samples and inverts the Δ value, adjusting the sampling interval appropriately.

4. *Use of Digital Filtering*

The use of digital filtering (1D and 2D) is a very useful approach for data compression, for several reasons (see also Section V). First, if the useful information is concentrated in a limited frequency band, digital filtering can extract this band, in particular through low-pass or bandpass filtering; indeed, a lower number of data are required to represent the extracted limited band (in comparison with the overall spectral extension) and hence a data compression result is obtained.

Further low-pass digital filtering is useful in preprocessing before the application of particular data compression methods, because the smoothed data can be more efficiently compressed by specific compression algorithms.

The joint use of digital filtering and data compression methods is presented in detail in Section V.

5. *Use of Transformations*

Orthogonal transformations, such as Fourier, Hadamard–Walsh, Haar, Karhunen–Loeve, etc., in particular in discrete or digital form, can be used for data compression, due to the fact that in general they represent a more compact representation of image data. This means that the transformed data become defined and exist in a smaller region or domain than the original data; a lower number of significant transformed data then result.

The 2D discrete Fourier transform (DFT) is defined as in Pratt (1978)

$$F(k_1, k_2) = \sum_{n_1=0}^{N-1} \sum_{n_2=0}^{N-1} f(n_1, n_2) \exp\left(-\frac{2\pi j}{N}(k_1 n_1 + k_2 n_2)\right) \tag{61}$$

while the inverse discrete Fourier transform (IDFT) is expressed as

$$f(n_1, n_2) = \frac{1}{N^2} \sum_{k_1=0}^{N-1} \sum_{k_2=0}^{N-1} F(k_1, k_2) \exp\left(\frac{2\pi j}{N}(k_1 n_1 + k_2 n_2)\right) \tag{62}$$

where the k_1, k_2 indices correspond to frequencies ($\nu_1 = k_1 \Delta\nu$, $\nu_2 = k_2 \Delta\nu$, $\Delta\nu$ being a constant-space-frequency interval).

With suitable symmetry properties, the discrete cosine transform and sine transform (DCT and DST, respectively) can be used. The DCT transform can be expressed in the following way

$$F(k_1, k_2) = 2 \sum_{n_1=0}^{N-1} \sum_{n_2=0}^{N-1} f(n_1, n_2) \cos\left(\frac{k_1 \pi}{N}(n_1 + \tfrac{1}{2})\right) \cos\left(\frac{k_2 \pi}{N}(n_2 + \tfrac{1}{2})\right) \quad (63)$$

The Hadamard transform is based on the properties of the Hadamard matrix (square form with elements equal to ± 1, having orthogonality between the rows and columns). A normalized Hadamard matrix, of $N \times N$ size, satisfies the relation

$$\underline{H}\underline{H}^T = 1 \quad (64)$$

The orthonormal Hadamard matrix of lowest order is the 2 × 2 Hadamard matrix

$$\underline{H}_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad (65)$$

The above transform is also known in the litearture as a Walsh transform (WT). A frequency interpretation of the above Hadamard matrix is indeed possible: the number of sign changes along any Hadamard matrix row, divided by 2, is called the sequence of the row. The rows of a Hadamard matrix of order N can also be considered as samples of rectangular functions having a subperiod equal to $1/N$; these functions are called Walsh functions (Pratt, 1978).

The Haar transform is based on the Haar matrix, which contains elements equal to ± 1 and 0.

One of the most efficient transforms is represented by the Karhunen–Loeve transform (KLT), which can be defined in the following way

$$F(k_1, k_2) = \sum_{n_1=0}^{N-1} \sum_{n_2=0}^{N-1} f(n_1, n_2) A(n_1, n_2; k_1, k_2) \quad (66)$$

where the $A(n_1, n_2; k_1, k_2)$ kernels satisfy the relation

$$\lambda(k_1, k_2) A(n_1, n_2; k_1, k_2) = \sum_{n'_1=0}^{N-1} \sum_{n'_2=0}^{N-1} C(n_1, n_2; n'_1, n'_2) A(n'_1, n'_2; k_1, k_2) \quad (67)$$

where $C(n_1, n_2; n'_1, n'_2)$ denote the covariance function of image data and $\lambda(k_1, k_2)$ are constants (eigenvalues of covariance functions) for fixed values of k_1 and k_2.

The above-considered discrete transforms can in general be evaluated in a fast form; the computing operation is divided in a sequence of subsequent computing steps in such a way that the results of the first computing steps (partial results) can be utilized repetitively in subsequent steps. Efficient software packages are available for Fast Fourier Transforms (FFT) and Fast Walsh Transforms (FWT). For instance, the number of operations required to evaluate 1D FFT becomes $N \log_2 N$ instead of N^2 for DFT, and to evaluate 1D FWT, $N \log_2 N$ instead of N^2 for DWT (the difference between DFT–FFT and DWT–FWT is that for the first type of transform the operations are complex multiplications and additions, while for the second the operations are additions and subtractions).

As already outlined above, the transformed data constitute a compact representation of the original image data; the number of significant transformed data is appreciably smaller than the number of original image data. For instance, an image constituted by a regular smoothed variation of grey levels will be represented by few Fourier components (few FFT data), while an image containing sudden variations in the grey level (nearly rectangular grey-level variations along the rows and columns) will be represented by few Walsh components (few FWT data).

Further, the transformed data can be compressed in a stronger way by applying simple algorithms such as thresholding (for instance, setting to zero the values under a small threshold such as a few percent) or prediction interpolation. The block diagram of a system applying this last approach, in particular to verify the efficiency of thresholding or the ZOP algorithm also through suitable displays, is shown in Fig. 7.

Variable-word-length coding can also be used for different transformed data blocks. In practice, the transformed data are divided into several squares and a minimum word length (a bit number sufficient to represent the maximum absolute amplitude value in the square plus 1 bit for the sign) is employed for each of them. In the actual storing or transmission of the processed image data, an additional fixed-length word is inserted before each square data, in order to specify the number of bits to represent the square coefficients.

With reference to 2D FWT, if $N = 2^n$, with n an integer, is the number of rows and columns of the sampled image and L is the number of grey levels, the maximum value which the transform will assume (corresponding to the addition of all the image samples) will be $N^2 L$. If q is the quantization value for the transformed data, the number of bits required to specify the word length used in a square will be

$$n_b = \log_2[\log_2(N^2 L/q + 1)] \quad (68)$$

where the algorithms are rounded to the next highest integer.

A modification of the above method for image data compression by means of 2D FFT or FWT consists in applying the same procedure of variable-word-length coding of the transformed data in a limited number of transformed

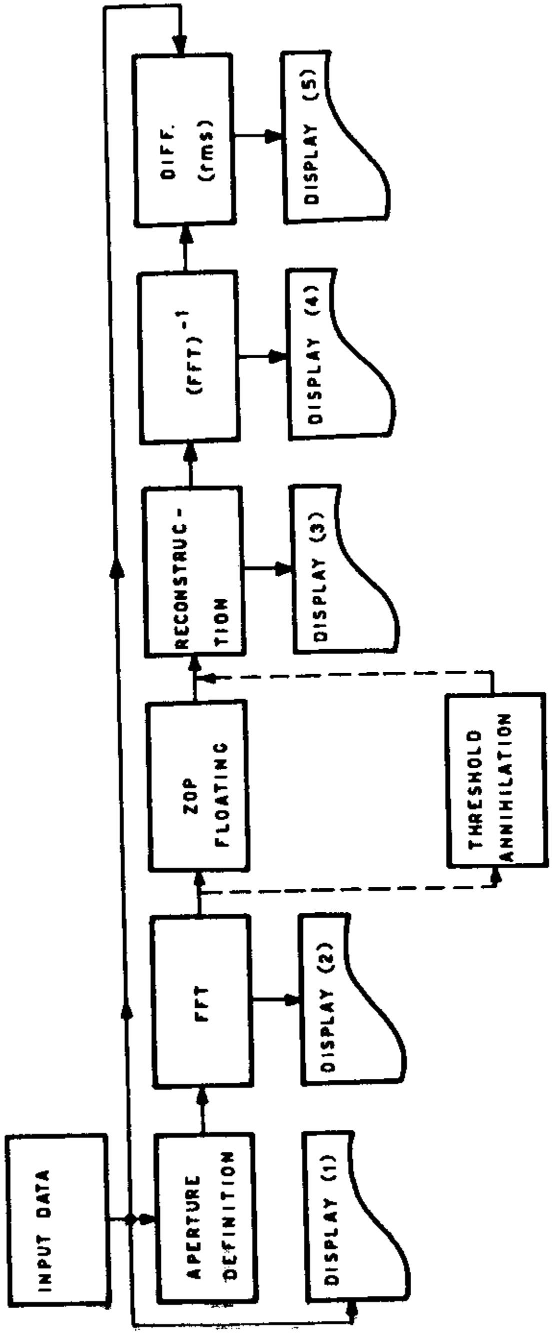


FIG. 7. Block diagram of a data compression system using FFT with thresholding or a ZOP algorithm.

image subareas. In particular no value can be maintained for those subareas, where the addition of the absolute values of the transformed data is below a given threshold.

From a computational viewpoint, we can further outline the following comparison considerations:

(1) 2D FFT is in general more efficient for images having continuous regular or smoothed variations (sine-wave type) in the grey level;
(2) 2D FWT is more efficient for images having sudden variations (of a rectangular type) in the grey level;
(3) Karhunen–Loeve transform is the most efficient, but it is much more complex than the others, and no fast computing routine is available for its use.

V. Joint Use of Two-Dimensional Digital Filters and Data Compression

The above-considered 2D digital filtering and data compression operations can be joined together with significant advantages for digital image-processing efficiency. The two operations are in general performed in cascade, one after the other.

Most parts of images indeed require some kind of filtering to smooth the data or to perform space-frequency corrections and obtain enhancement, in general with the goal of reducing the noise or disturbances and of obtaining higher-quality images. Data compression is, as already outlined, a desirable operation after filtering to reduce the amount of data, which is becoming a tremendous problem for the practical use of images in many application areas.

Further, the combination of the two operations can be attractive to increase the compression efficiency (see Section IV,B,4); smoothed data after low-pass filtering can surely be more efficiently compressed by the different data compression algorithms, because they now operate on 2D data having lower space-frequency values.

In the following some typical connections of the two digital operations are first presented; then a special new system, based on digital filtering and data reduction, is described for digital comparison and correlation of digital images having different space resolutions.

A. Some Typical Connections of the Two Digital Operations

Local space operators, 2D digital filters, and data compression can be connected in different useful ways to obtain some specific results on the processed image.

(1) Low-pass filtering (by means of a local space smoother or 2D digital filtering) and thresholding: high space-frequency components can be reduced due to the noise and disturbances, and hence a binary image can be obtained, where the more useful data are maintained (by selecting suitable threshold values).

(2) High-pass filtering (by means of local differential operators or 2D digital filtering) and thresholding: image enhancement is performed, giving higher contrast to image structures (grey-level variations), and hence a binary image is obtained, where some useful structures and patterns can be extracted (by selecting suitable threshold values).

(3) Low-pass filtering [as in (1)] and compression by means of prediction interpolation, DPCM, DM, or variable-word-length coding: high space-frequency noise is reduced, and in the meantime smoothed data are more efficiently compressed.

(4) Low-pass filtering [as in (1)] followed by edge detection and hence spike elimination (by means of nonlinear operators as in Section III,A): high space-frequency noise is reduced (for instance, random noise), useful edges and boundaries are extracted (representing a compressed form of the image), and further high-amplitude spikes or scintillation noise are eliminated.

(5) Low-pass filtering [as in (1)] and compression by means of digital transformations (2D FFT or FWT): the same result as in (3) can be obtained.

(6) Use of 2D FFT to perform filtering and compression: once the 2D FFT has been evaluated, high space-frequency components can be discarded to obtain a filtering effect (smoothing with noise reduction); hence the remaining 2D FFT components can be reduced with thresholding or variable-word-length coding (see Section IV,B,5).

B. *Processing System for Digital Comparison and Correlation of Images Having Different Space Resolution*

In many application areas, such as remote sensing, biomedicine, and robotics, an important practical problem is represented by the availability of several images given by different sensors or equipment and regarding the same scene (land region, body organ, mechanical object, ...). In general these images are taken from different view points and have different space resolution. Increasingly often a processing goal is to obtain integrated images or maps, where the data from the different images pertaining the same observed scene are suitably correlated (for instance, through a simple addition difference or specific weighting of one image's data by the other images' data).

To solve the above problems, there are two types of digital processing to be performed:

(1) geometrical corrections and rotations with a change of viewpoint (to refer the different images to the same viewpoint or to the point at infinite distance and orthogonal position producing orthoimages);

(2) space resolution variations in such a way finally to have images with the same space resolution, which can be actually integrated.

While for point (1) there are several geometrical transformations available using trigonometric functions, for point (2) there are few approximation procedures. In the following a rigorous method is presented, based on 2D digital filtering and data reduction (Cappellini *et al.*, 1984a).

Let us consider two images $f_1(n_1, n_2)$ and $f_2(n_1, n_2)$ in digital form, the first with high space-frequency resolution or definition and a space-sampling interval X_1, the second with lower space-frequency resolution and a sampling interval $X_2 > X_1$. Practically, if $m = (X_2/X_1)$, to one pixel of the image $f_2(n_1, n_2)$ corresponds m^2 pixels of the image $f_1(n_1, n_2)$.

Several approaches can be used to obtain from a high-definition image $f_1(n_1, n_2)$ an image $g_1(n_1, n_2)$, having a space-sampling interval X_2 equal to that of the lower-definition image [$g_1(n_1, n_2)$ is a compressed form of $f_1(n_1, n_2)$].

One simple technique corresponds to evaluating $g_1(n_1, n_2)$ data as the usual average of $f_1(n_1, n_2)$ data [see also Eq. (24)]; that is (with m odd),

$$g_1(n_1, n_2) = \frac{1}{m^2} \sum_{k_1 = -(m-1)/2}^{(m-1)/2} \sum_{k_2 = -(m-1)/2}^{(m-1)/2} f_1(n_1 - k_1, n_2 - k_2) \qquad (69)$$

The $g_1(n_1, n_2)$ image obtained in this way represents a rough smoothed version of the original high-resolution image $f_1(n_1, n_2)$.

A second, more refined approach consists in evaluating $g_1(n_1, n_2)$ data as a weighted average of $f_1(n_1, n_2)$ data; that is,

$$g_1(n_1, n_2) = \sum_{k_1 = -(m-1)/2}^{(m-1)/2} \sum_{k_2 = -(m-1)/2}^{(m-1)/2} w_a(k_1, k_2) f_1(n_1 - k_1, n_2 - k_2) \qquad (70)$$

The weights $w_a(k_1, k_2)$ in the above relation define the form of smoothing operation which is performed on the $f_1(n_1, n_2)$ data; it is easy to verify, for instance, that with $w_a(k_1, k_2) = (1/m^2)$ Eq. (70) is equivalent to Eq. (69). It can appear reasonable to give, in general, greater weight to the central pixels of the $m \times m$ subimage of the $f_1(n_1, n_2)$ image with respect to the peripheral ones. For this purpose one solution corresponds to using a linear weighting resulting in a conical (or pyramidal) function in the 2D domain; another solution consists in using a Gaussian weighting function (the 2D function can be easily obtained through the circular rotation of a 1D Gaussian function).

The above-described techniques perform a smoothing operation on the high-resolution image $f_1(n_1, n_2)$ in a heuristic way to obtain the image

$g_1(n_1, n_2)$ to be compared and correlated with the lower-resolution image $f_2(n_1, n_2)$. A more rigorous and precise system is based on the use of a 2D digital filter of the low-pass type with circular symmetry (see Section II). The precise steps of this system are the following:

(1) to perform a low-pass, circular symmetry, 2D digital filtering with a cutoff frequency $\omega_c/2\pi = 1/2X_2$, obtaining the filtered image $g_1(n_1, n_2)$;

(2) to reduce or "decimate" the obtained data $g_1(n_1, n_2)$ up to a space-sampling interval equal to X_2, that is, to obtain the image (in digital form) $g_2(n_1, n_2) = g_1(n_1 X_2, n_2 X_2)$.

The above digital operations indeed remove from the 2D spectrum of the high-resolution image $f_1(n_1, n_2)$ the space-frequency components greater than $\omega_c/2\pi$, giving therefore an image which is directly comparable—for that which regards the space resolution—to the lower-resolution image $f_2(n_1, n_2)$. The two digital images $g_2(n_1, n_2)$ and $f_2(n_1, n_2)$ now result to have grey-level variations in the different space directions with the same maximum space frequency.

It is interesting to observe that the 2D digital filter used in this last rigorous system includes, as particular cases, the approaches defined by Eqs. (69) and (70). The first is obtained by setting the coefficients of the 2D digital filter $a(k_1, k_2) = (1/m^2)$ [see Eqs. (2) and (9) for the FIR case]; the second results by setting $a(k_1, k_2) = w_a(k_1, k_2)$.

VI. Applications

In the following some examples of applications of 2D digital filters, local space operators, and data compression to such important fields as communications, remote sensing, biomedicine, and robotics are presented.

A. *Applications to Communications*

In a communications system a message is transmitted from one place to another through different physical communication channels (lines, cables, satellite links, optical fibers, etc.). If the transmitted message is a signal $s(t)$ (Fig. 8), the receiver produces an estimate $s_e(t)$ of the original source message, trying to reduce the noise and degradation introduced by the channel.

Often many messages of the same or different type have to be sent in parallel to utilize the communications medium in a more efficient way. Multiplex communication systems are used for this purpose. Two important types of multiplex systems are represented by frequency division multiplex

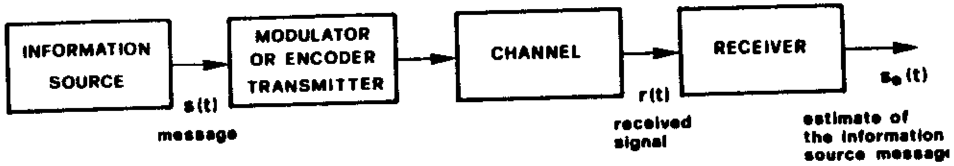

INFORMATION SOURCE
MODULATOR OR ENCODER TRANSMITTER
CHANNEL
RECEIVER
s(t)
message
r(t)
received signal
$s_e(t)$
estimate of the information source message

FIG. 8. General block diagram of a communication system.

(FDM) and time division multiplex (TDM). In the first the single messages are set in adjacent frequency domain, while in the second the messages are organized in subsequent time intervals, in general by sending one sample of each message after the other in a cycle or frame and sending one frame after the other. By representing each sample in the TDM system in digital form (a word of a given number of bits) a pulse-code-modulation (PCM) multiplex system is obtained.

Digital communications of the PCM type have expanded recently in an exponential manner, due to the main aspects outlined in the Introduction (Section I). In digital communications it is easy to apply digital operations such as previously described (digital filtering, local space operators, data compression).

By considering the transmission of images, these are in general converted into a video signal through a scanning procedure and then this signal is sampled and set in digital form. Digital operations described in the previous Sections can be usefully applied to this digital signal both in transmission and reception, according to the general block diagram in Fig. 9.

In transmission 1D digital filtering can be performed to reduce high-frequency noise components and exactly define the bandwidth. 2D digital filtering and local space operators can also be applied, if a suitable memory or buffer is available, processing image data in such a way as to reduce high space-frequency components (image smoothing) or to obtain image enhancement. Hence data compression can be performed to maintain the more significant data. By means of filtering and compression, a bandwidth reduction or bandwidth compression is achieved, which is a very important result to increase the efficiency of the digital communication system (the same image data can be transmitted by using a smaller bandwidth, or other data can be transmitted with the image by using the same bandwidth). A channel coder can be added after compression to protect the remaining important data against channel noise and disturbances (Benelli *et al.*, 1977, 1984).

In reception, after the eventual channel decoder for error detection and correction, the image data are reconstructed (decompression) also utilizing synchronization data (given by a sequence detector) and hence digital filtering is performed (1D and 2D type) to reduce remaining channel noise or to

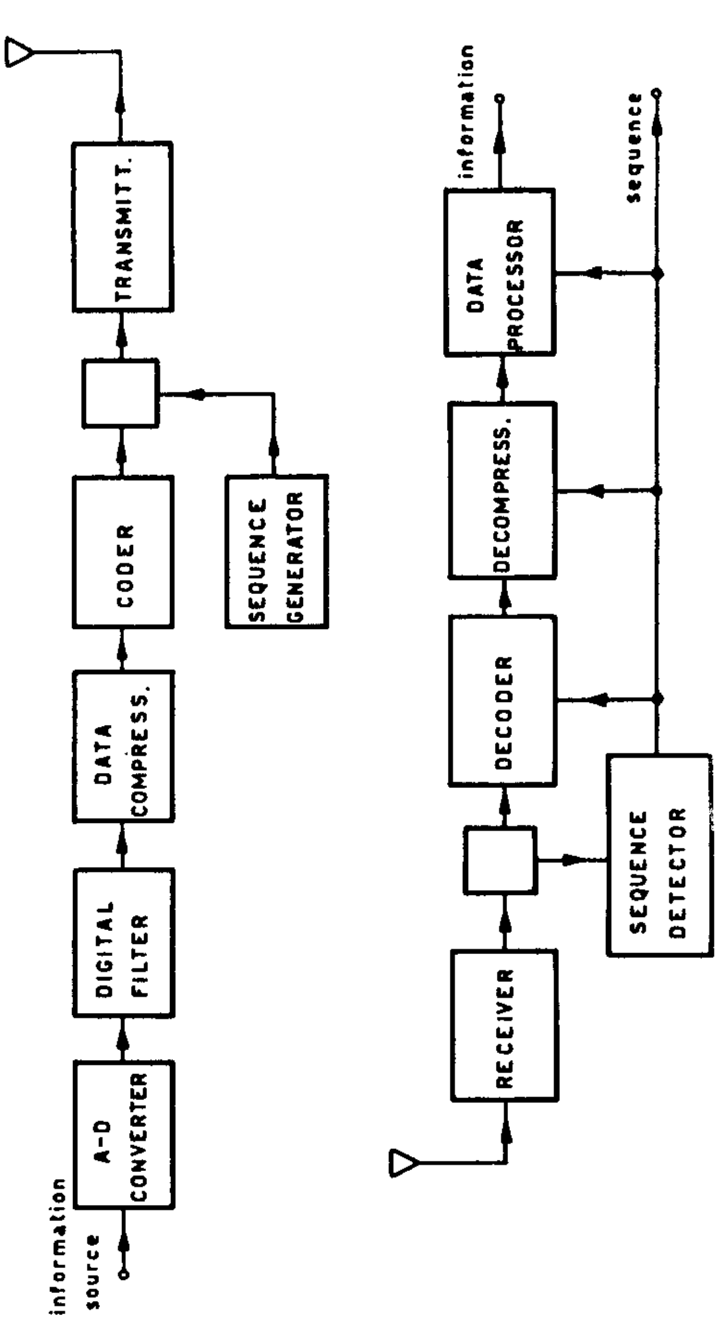


FIG. 9. Block diagram of a digital communication system, using digital filtering, data compression, and error-controlled coding operations.

increase image quality. With reference to this last filtering processing, 1D and 2D digital filters can indeed be applied in the following way:

(1) 1D digital filtering can reduce channel noise and degradation (such as multipath and Doppler effects) through channel equalization (fixed and adaptive type) and matched filtering (Cappellini *et al.*, 1978);

(2) 2D digital filtering can reduce space-frequency components due to noise and perform image restoration (inverse filtering) and enhancement.

Let us now consider in more detail the digital transmission of time-varying images (television) and of time-fixed or static images.

In the first case, for good reproduction of movement, image sequences are required at a sufficient rate. In European TV standards for instance, 25 images/s are used. By using 625 lines/image and 8 bits/sample, transmission rates of 50 Mbits/s are obtained. To reduce this high value, the analysis of two subsequent images can be performed in such a way as to take into account, for instance, only the actually moved parts in the transition from one image to the subsequent one (interframe techniques). DPCM techniques (see Section IV,B,3) can be used for this purpose; through variable-word-length coding, mean word lengths of 2–2.5 bits/sample are obtained, reducing the transmission rate to 10–20 Mbits/s. A suitable encoding, as with prediction interpolation or—more efficiently—with digital transformations (see Section IV,B), can also be performed in each single image (intraframe techniques). Combining interframe and intraframe techniques, lower bit rates (2–10 Mbits/s) are obtained, at the expense of higher complexity and cost.

An interesting approach to reduce the redundancy in TV images, by means of inter–intraframe coding, is based on movement compensation: The coding consists essentially in determining for each pixel the prediction model (spatial, temporal) and transmitting, when necessary, the quantized difference and the prediction model changes (Brofferio *et al.*, 1975).

In the case of videotelephone or teleconference systems, due to the lower number of images points in movement from one image to another, and by using DPCM techniques with variable-word-length coding, 0.9–1 bits/sample are sufficient. If information about image moving objects or parts is suitably used, values of 0.4–0.5 bits/sample are reached. For instance, for an object translation, the shift and direction values can be sent to the receiver, which will also reconstruct grey levels of moved image parts. Transmission rates on the order of a few Mbits/s are thus obtained.

In the second case of static images, two practical situations can be considered: transmission of written documents and sheets, and transmission of photos (telephoto).

Regarding documents and sheets, let us consider a standard A4 sheet (29.6 × 20.8 cm). To describe the written information and transmit it to the receiver (as in facsimile), an efficient method can be represented by the use of an optical reader of the written characters. If the sheet contains 30 lines, each with 70 characters, there are 2100 characters in a sheet. If, for instance, 7 bits are utilized for the transmission of the characters, 14,700 bits are required for the representation of the sheet. If, however, variable-word-length coding is used, taking into account the character probabilities (see Section IV,A), a lower number of bits is required (as an example, for the English language, a mean word length—equal to the source entropy—4.2 bits/character is resulting, with 8800 bits required to represent the sheet).

A more economic system can scan the sheet (assumed to be black characters on a white background) with 1200 lines (at least 4 lines/mm are required) and represent each line by 800 equidistant space samples (points). Each sample being of a binary value, there are 960 000 bits required to represent the whole sheet (an amount much greater than the previous one). A little more efficient coding is obtained through the representation of the lengths of black or white point sequences by means of variable-word-length coding: Mean values of 0.3–0.4 bits/point are sufficient. By using the variations of the above sequences line by line (comparing one line with the following one) and suitable encoding of these variations, mean values of 0.1–0.2 bits/point are obtained (with 96,000–180,000 bits required to represent the whole sheet).

For what concerns the representation and transmission of black and white photos, the number of data required is much higher. Let us consider a telephoto of 13 × 18 cm. To have sufficient space resolution, 8.6 lines/mm are required, and therefore 1500 lines with 1100 samples/line result. The overall amount of data, by representing the grey level of each sample by 7 bits, is therefore 11.5 Mbits (by using transmission with 4800 bits/s, 40 minutes are required for the transmission of a complete photo). By means of data compression techniques, such as DPCM with variable-word-length coding (see Section IV,B,3) a mean word length of 2–3 bits/sample is obtained, and by means of digital transformation (see Section IV,B,5) a value of 1–2 bits/sample and less can be reached. With reference to this last approach, Fig. 10 shows an example of the application of the discrete cosine transform (DCT, implemented in a fast way, FCT) to a typical photograph of Florence (the Old Bridge). Thresholding compression of transformed data is used, representing image square sub-blocks containing $N_s \times N_s$ data (transformed data less than 1% are neglected). Fig. 10a shows the original digitized photo, and Fig. 10b the reconstruction in the case $N_s = 16$ (mean word length = 0.6 bits/sample, $e_p = 16.86\%$, and $e_r = 2.13\%$).

Finally, it is important to observe that all the above data-compression and data-rate-reduction results can be improved, if 2D digital filtering or local

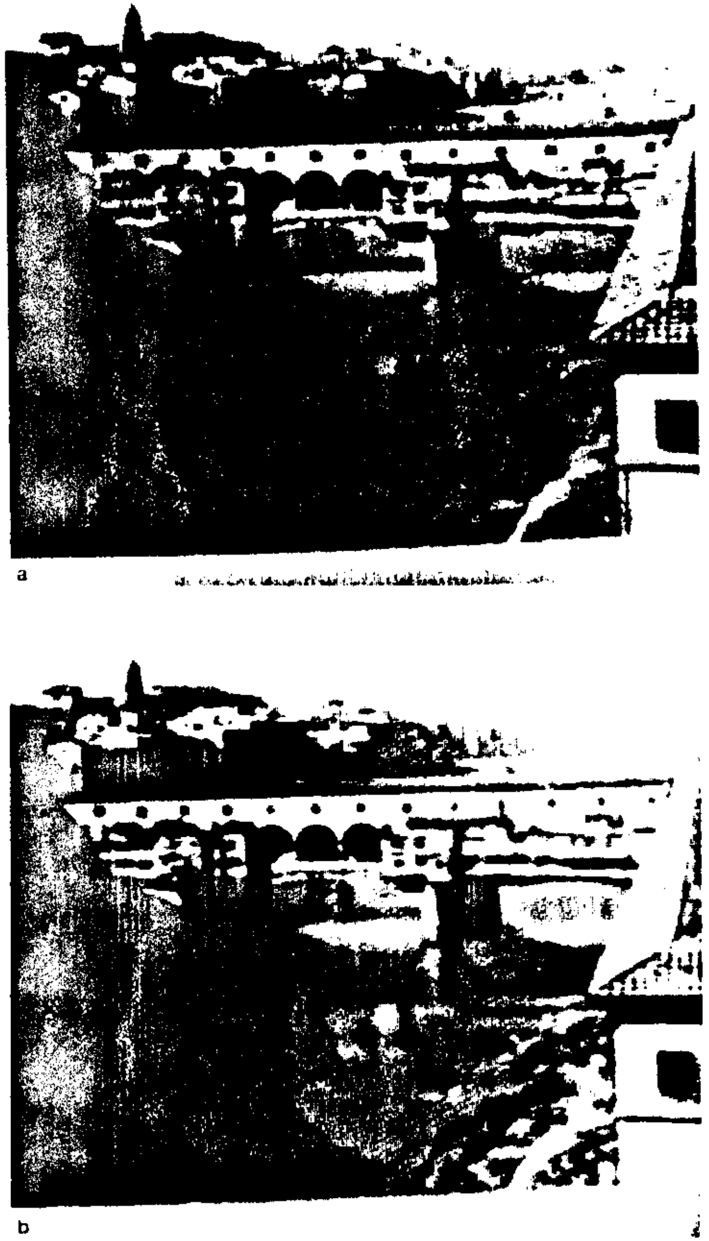


FIG. 10. Example of the application of the fast cosine transform (FCT) to perform data compression on a photograph of the Old Bridge in Florence: (a) original digitized photo; (b) reconstructed photo (with 0.6 bits/sample).

space processing (as of the low-pass filtering type) is performed before the application of data-compression techniques.

B. Applications to Remote Sensing

Many remote sensing images and maps are currently collected by platforms aboard aircrafts and satellites. Indeed passive remote sensing systems, as optical cameras, multispectral scanners (MSS), and microwave radiometers, or active remote sensing systems, as side-looking radar (SLR) and laser radar (lidars), give an impressive amount of images and data.

The above images, maps, and data given by remote sensing systems in general need to be processed to improve their quality (geometric and sensor corrections, noise reduction, enhancement,...) and to obtain final useful results (extraction of specific regions and land–sea areas as for agriculture investigation or water resource monitoring). 2D digital filters, local space operators, and data compression represent indeed very useful digital operations to achieve the above outlined goals.

2D digital filters or local space operators can be applied as a preprocessing operation to smooth the image data (by means of low-pass filtering) or to perform a space-frequency correction or to obtain enhancement (by means of high-pass or bandpass filtering), also extracting edges and boundaries. In particular, after enhancement better-quality images can, in general, be obtained and through edge extraction different earth regions can be recognized and classified with easy evaluation of the corresponding areas. Data compression can be applied in general after some type of filtering, to reduce the amount of data, which is becoming a tremendous problem for the practical use of satellite data and aircraft photos of large earth areas (Cappellini, 1980). In the following some typical processing examples are given, regarding the application of the above digital operations.

Figure 11 shows an example of the application of filtering and edge extraction to an aircraft photo (a region south of Florence). In Fig. 11a the digitized image is shown, while in Fig. 11b the result of processing by means of the nonlinear filtering operator as defined by Eq. (26) (nonlinear smoother) followed by the isotropic-gradient edge detector [Eq. (37)]. As it appears, the main different ground regions are isolated; adding the grey-level information, three classes can easily be obtained: forest (black and high-intensity grey levels), wine grapes and oil plants (medium-intensity grey levels), and other ground regions.

Another very simple processing example is shown in Fig. 12. A LANDSAT-C image of the Tirrenic coast (at the bottom the Arno River appears) is processed first through grey-level expansion (stretching, see Section III) and then through thresholding. Figure 12a shows the original image, and Fig. 12b gives the final result, which practically corresponds to an estimation of the water resources in the analyzed region.

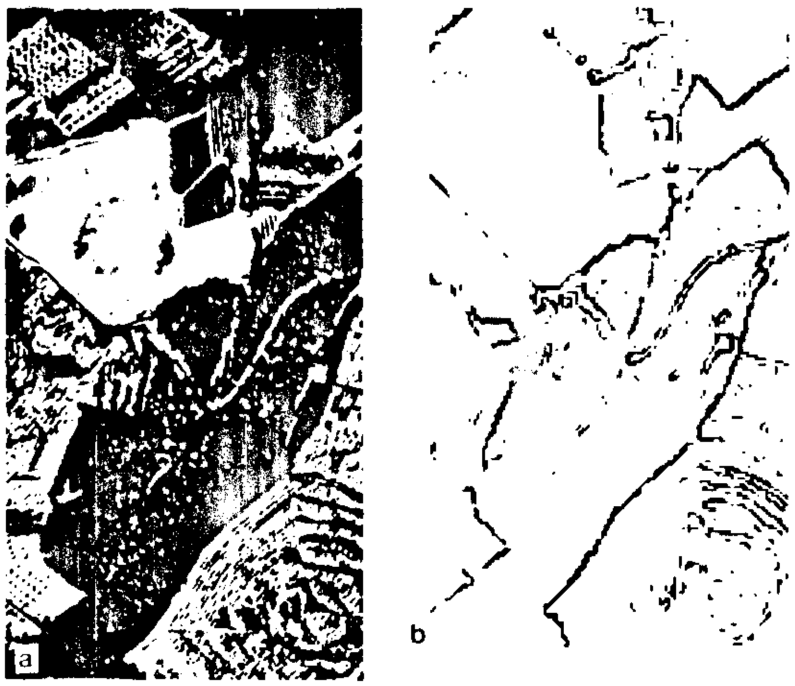


FIG. 11. Example of the application of nonlinear smoothing and edge extraction to an aircraft photo (region south of Florence): (a) digitized photo; (b) processed result.

Figure 13 shows an example of the application of a 2D FIR digital filter of the high-pass type [Eq. (2)] to a LANDSAT-C image. At the right is a part of the original image (North Africa), while at the left the filtered image appears. As is clear, a good enhancement effect results, which can be very useful for extracting some significant regions; further, through thresholding as in Fig. 12b, final estimate regarding these regions could be obtained (Cappellini, 1984).

Figure 14 shows an example of the application of data compression with a ZOP algorithm and floating tolerance (see Section IV,B,2) to an ERTS-1 image. At the left the original image is shown, and at the right the reconstructed one after compression (an average compression ratio $C_{r_a} = 1.56$ is obtained).

Figure 15 gives another example of the application of data compression on the same ERTS-1 image, using the 2D FWT with variable-word-length coding

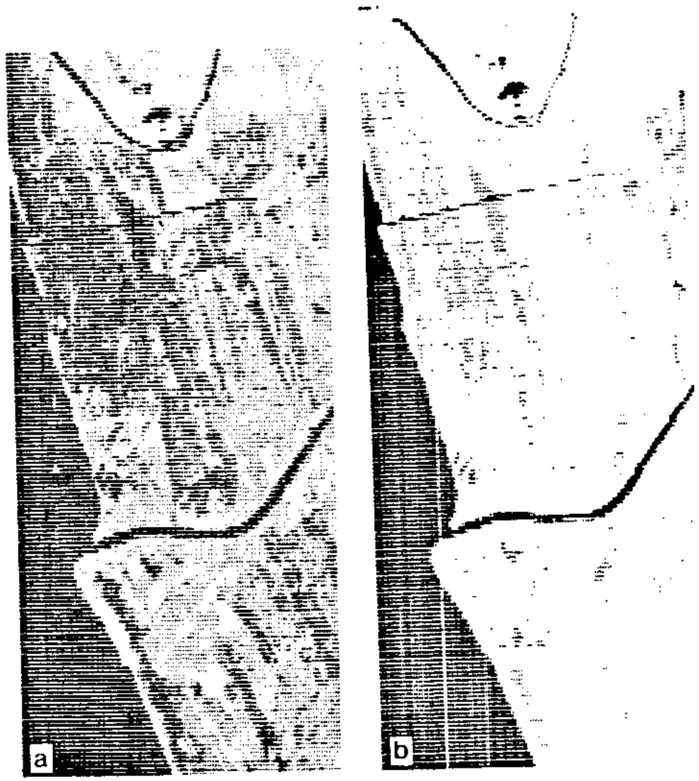


FIG. 12. Example of application of stretching and thresholding to a LANDSAT-C image (Tirrenic coast with the Arno River at the bottom): (a) original image; (b) processed result.

of transformed data blocks (4 × 4). At the left is the original image, in the middle the reconstructed one with $q/N = 4\%$ [see Eq. (68)] corresponding to a compression ratio $C_{rs} = 2.14$, and at the right the reconstructed one with $q/N = 8\%$ and $C_{rs} = 3.84$. Higher compression ratios can be obtained by increasing q/N (Cappellini *et al.*, 1976).

A practical application of the processing system for digital comparison and correlation of images having different space resolution, presented in Section V,B, is shown in Figs. 16–20. Figure 16 shows a SEASAT-SAR image (256 × 256) of a coastal region in South Italy (Sele River in Campania), which represents the high-resolution image $f_1(n_1, n_2)$. Figure 17 gives a LANDSAT-C image (256 × 256) of the same region, the image representing the lower-resolution one $f_2(n_1, n_2)$. Figure 18 shows the result of 2D FIR digital filtering (low-pass type, circular symmetry) of the SEASAT image. Figure 19 shows the two final images obtained for comparison and correlation. At the left is the LANDSAT image (a part of the original image suitably rotated to be registered with the SEASAT image); at the right is the SEASAT filtered image already decimated [corresponding to $g_2(n_1, n_2)$]. Figure 20 finally gives a simple integration test: the addition of the two images in Fig. 19 (Cappellini *et al.*, 1984a).

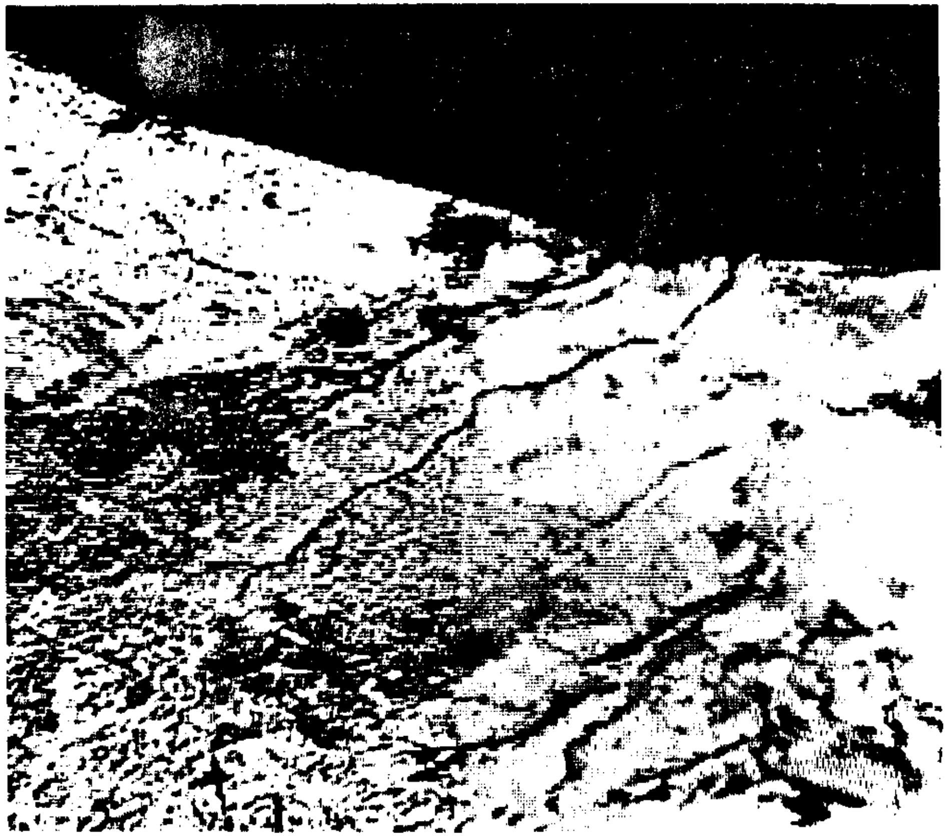


FIG. 13. Example of the application of a 2D FIR digital filter of the high-pass type to a LANDSAT-C image (North Africa): at the right is a part of the original image, while at the left the filtered image is emerging.

C. *Applications to Biomedicine*

With recent rapid technological evolution, much equipment has been introduced in biomedicine to produce different types of biomedical images or

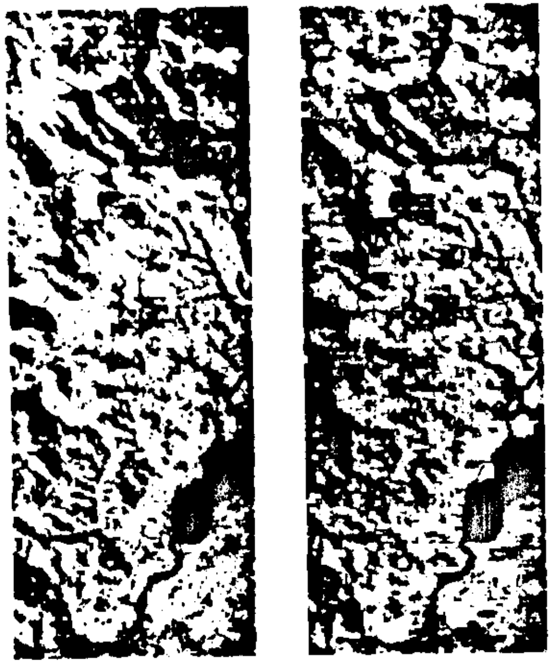


FIG. 14. Example of the application of data compression with a ZOP algorithm and floating tolerance to an ERTS-1 image: at the left the original image is shown, while at the right the reconstructed one is shown ($C_{rs} = 1.56$).

bioimages. Some examples of biomedical branches giving bioimages are: radiography (x-ray), thermography (ir), scintigraphy (nuclear medicine), ecography (ultrasonics), electrocardiography (ECG maps), electroencephalography (EEG maps), and computer tomography (CT). Other bioimages of increasing interest are nuclear-magnetic-resonance (NMR) images and microwave-radiometry images.

The above bioimages can be processed by 2D digital filters, local space operators, and data compression to obtain several useful results. By means of low-pass filtering, a smoothing of the bioimage is obtained, reducing high space-frequency noise components. By using high-pass or bandpass filtering, enhancement effects result, outlining and extracting useful data and patterns in other ways not clearly recognized. By means of inverse filtering (restoration), noisy bioimages can be processed to obtain higher-quality images for clinical diagnosis and interpretation. Data-compression techniques can hence reduce the amount of data, increasing in an impressive way, solving storage problems (archival systems), and increasing the efficiency of bioimage transmission from one place to another (telemedicine). In the following some typical examples are reported.

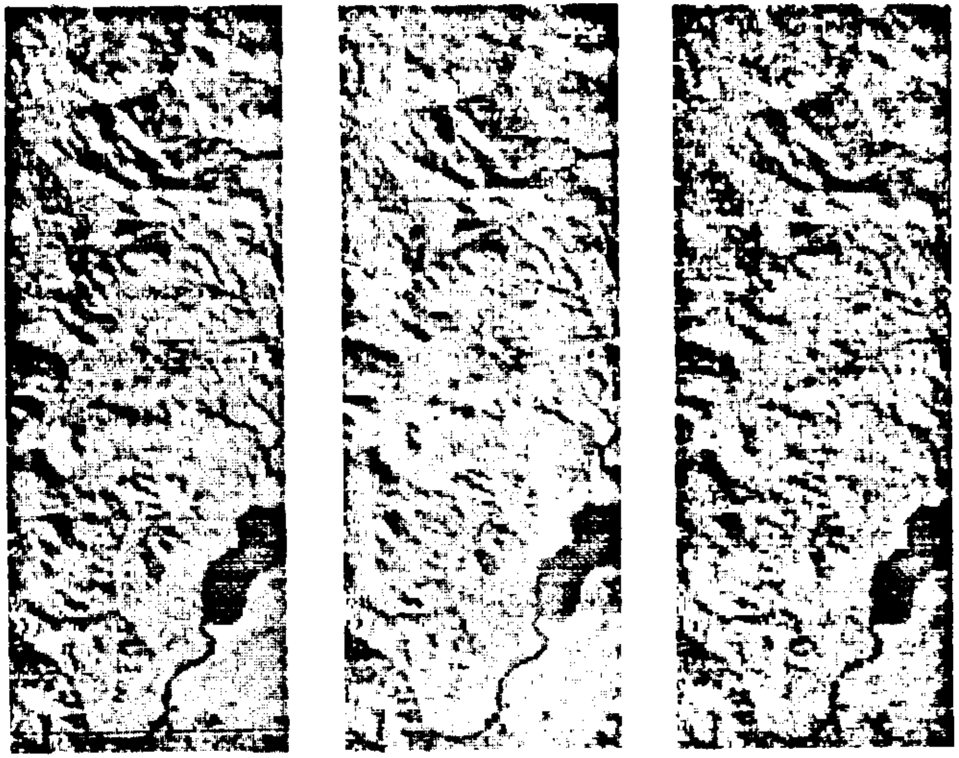


FIG. 15. Example of the application of data compression using a 2D FWT and variable-word-length coding of transformed data blocks (4 × 4) to an ERTS-1 image: at the left, the original image; in the middle, the reconstructed image ($C_{rs} = 2.14$); at the right, the reconstructed image ($C_{rs} = 3.84$).

A first example regards ECG or EEG maps. A special hardware system was recently built in Florence, containing a fast digital processor performing up to 256 1D digital filtering operations on ECG or EEG signals (Cappellini and Emiliani, 1983). In particular 16 signals (representing a 4 × 4 micromap) can be processed in realtime and the filtered data, for instance corresponding to α components in an EEG, can then be processed by computer systems performing 2D digital filtering or other compression operations. Indeed the 1D digital filtering performed on the 4 × 4 signals by the hardware system represents an interesting example of parallel processing of 2D data, extracting useful frequency components (and in this way performing also a sort of data compression). Figure 21 shows a standard chart recording of parallel filtering

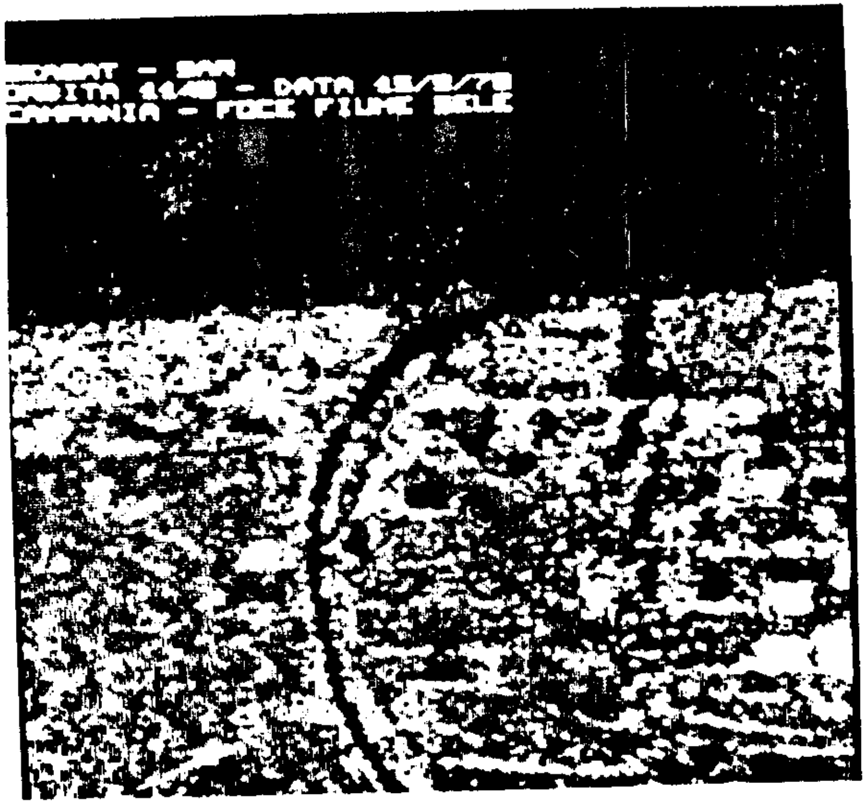


FIG. 16. A SEASAT-SAR image of a region in South Italy.

of four EEG signals (2 × 2 micromap), obtaining three outputs for any input signal in the frequency bands 8–10, 10–12, and 12–14 Hz.

Figure 22 shows an example of processing in infrared thermography. In Fig. 22a the original digitized image is given, in 22b the result of grey-level expansion (stretching) is reported in conjunction with edge detection applied on a limited range of grey levels, outlining the venous traces (Prosperi, 1983).

Figure 23 shows an example of the application of a 2D FIR digital filter of the bandpass type to a nuclear medicine image; at the top is the original image, and at the bottom the result of processing. Due to the special enhancement effect, a cyst now appears at the left of the image (the small black region).

Figure 24 shows another example of processing a computer tomography image. In 24a the original image is given, in 24b the result of linear stretching performed in the limited grey-level range 80–190, in 24c the result of a 2D FIR digital filtering of the parabolic type. As it appears this last filter can indeed be useful for obtaining special enhancement. It can be proved that a 2D parabolic filter (having flexible parameters as the origin and slope) is a good approximation of inverse or restoration filtering (Cappellini *et al.*, 1978). This example outlines how in computer tomography, in addition to the standard image manipulation provided, special effects can be obtained in particular by means of 2D digital filtering.

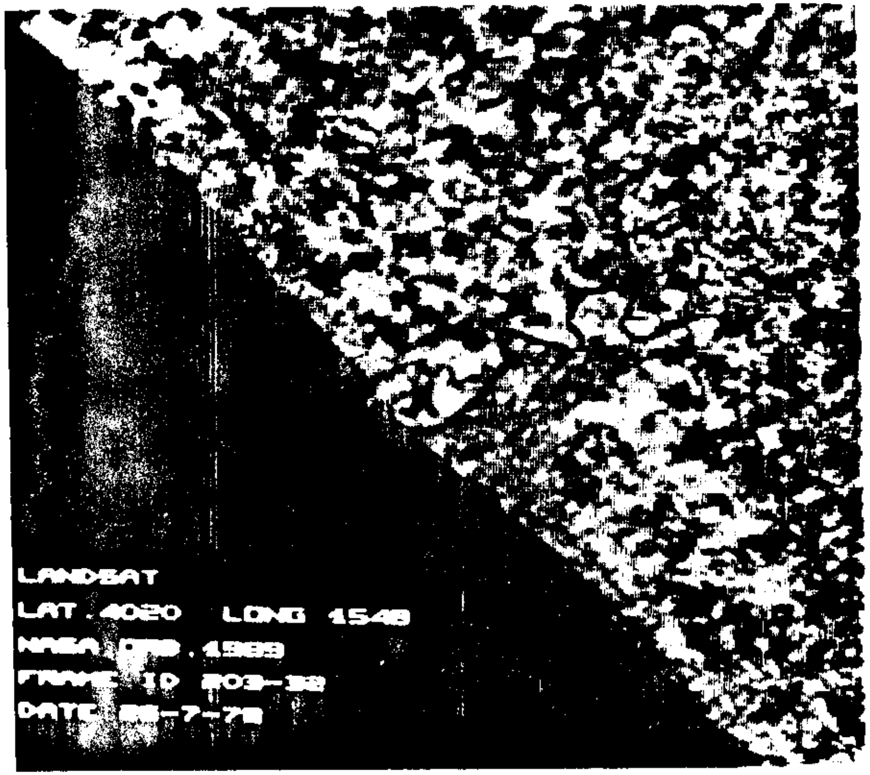


FIG. 17. A LANDSAT-C image of the same region as in Fig. 16 (extended area).

As already observed, 2D digital filtering can be very useful as preprocessing before data compression (see Section V). Table I shows some experimental results, obtained by processing nuclear medicine images before with 2D lowpass digital filtering and then with data compression using digital transformations (2D FFT and FWT). As it appears, with the same e_p and e_r errors, the compression ratio C_{ra} is appreciably increased when the 2D digital filter is

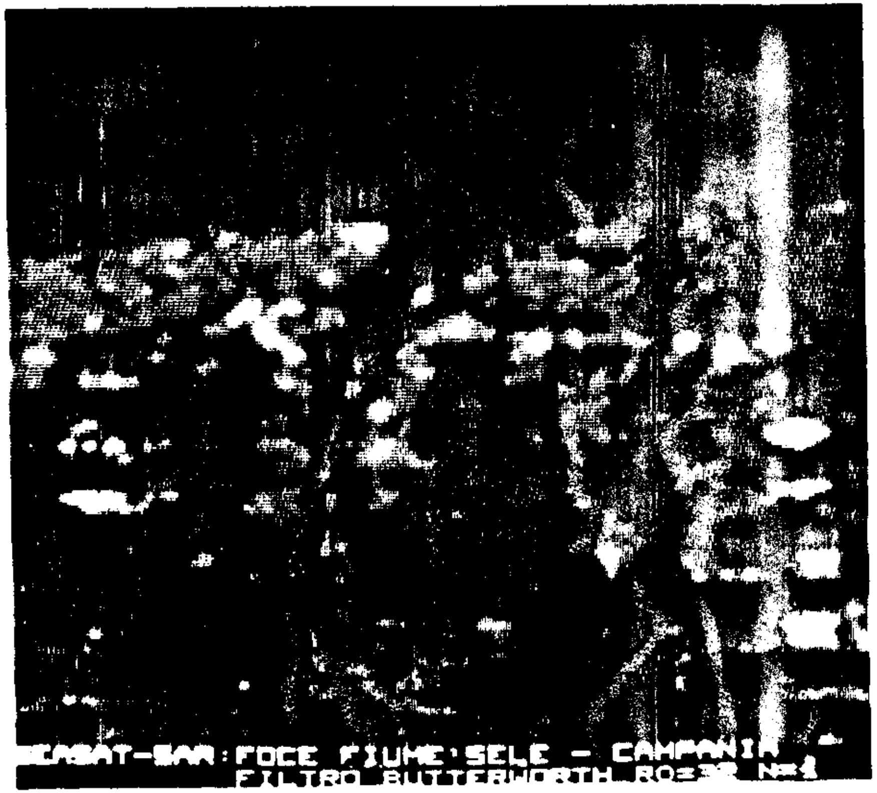

SEASAT-SAR: FOCE FIUME SELE - CAMPANIA
FILTRO BUTTERWORTH RO=32 N=1

FIG. 18. Result of 2D FIR digital filtering (low-pass type, circular symmetry) applied to the SEASAT image.

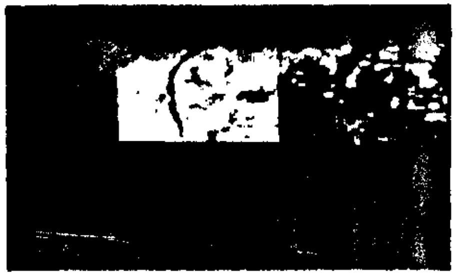


FIG. 19. The two final images obtained for comparison and correlation: at the left is the LANDSAT image; at the right, the SEASAT filtered and decimated image.

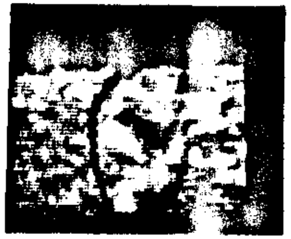
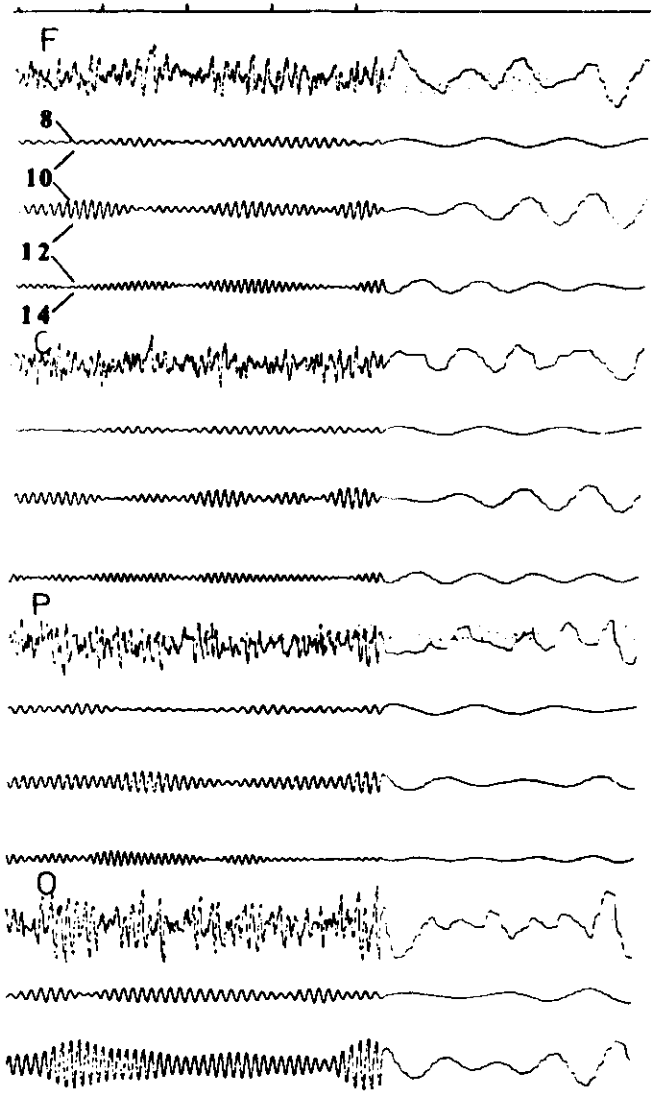


FIG. 20. A simple integration test: addition of the two images obtained in Fig. 19.



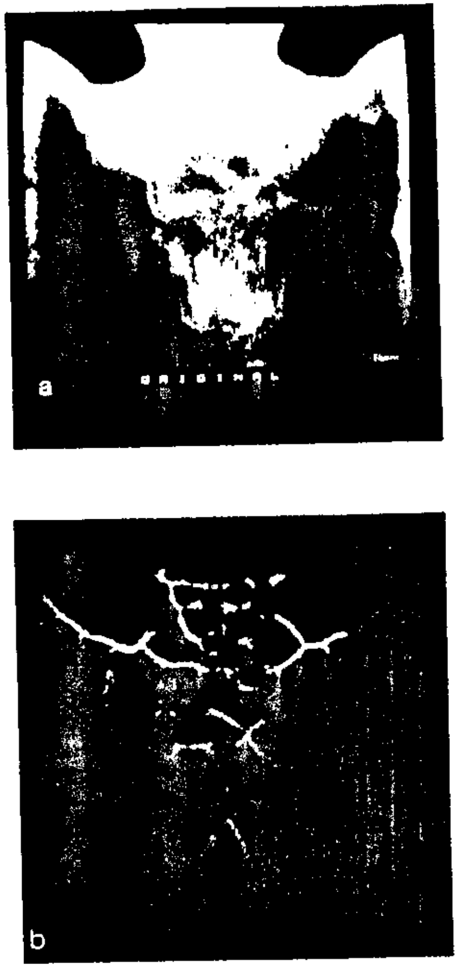


FIG. 22. Example of processing in infrared thermography: (a) original digitized image; (b) result of edge detection applied on a limited range of grey levels.

used in comparison with the situation with no prefiltering (in the first case C_r passes from 2.5 to 6, in the second one from 3.5 to 8) (Cappellini, 1979a).

D. *Applications to Robotics*

In computer vision for robotics, in which one or more scene sensors such as TV cameras take information on mechanical objects or other systems in a

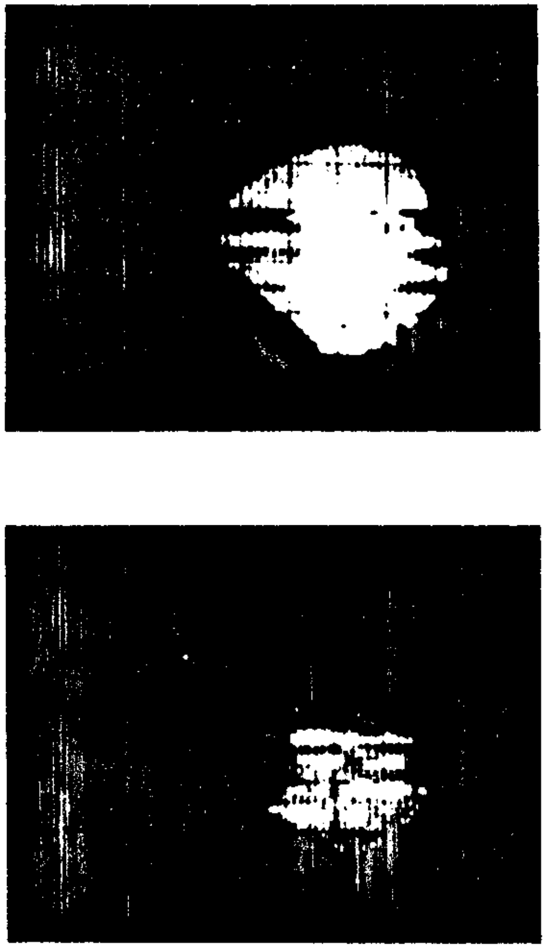


FIG. 23. Example of the application of a 2D FIR digital filter of the bandpass type to a nuclear medicine image: at the top is the original image; at the bottom, the result of processing.

static position or in movement, efficient processing techniques are required to analyze the images given by the sensors.

In particular, due to environmental noise conditions (light change, different colors of the objects, movement,...), preprocessing is required with fast filtering operations; hence edge detection is useful to extract the object's shape before final recognition and classification. In the following some examples of the application of digital operations described in the previous sections are given.

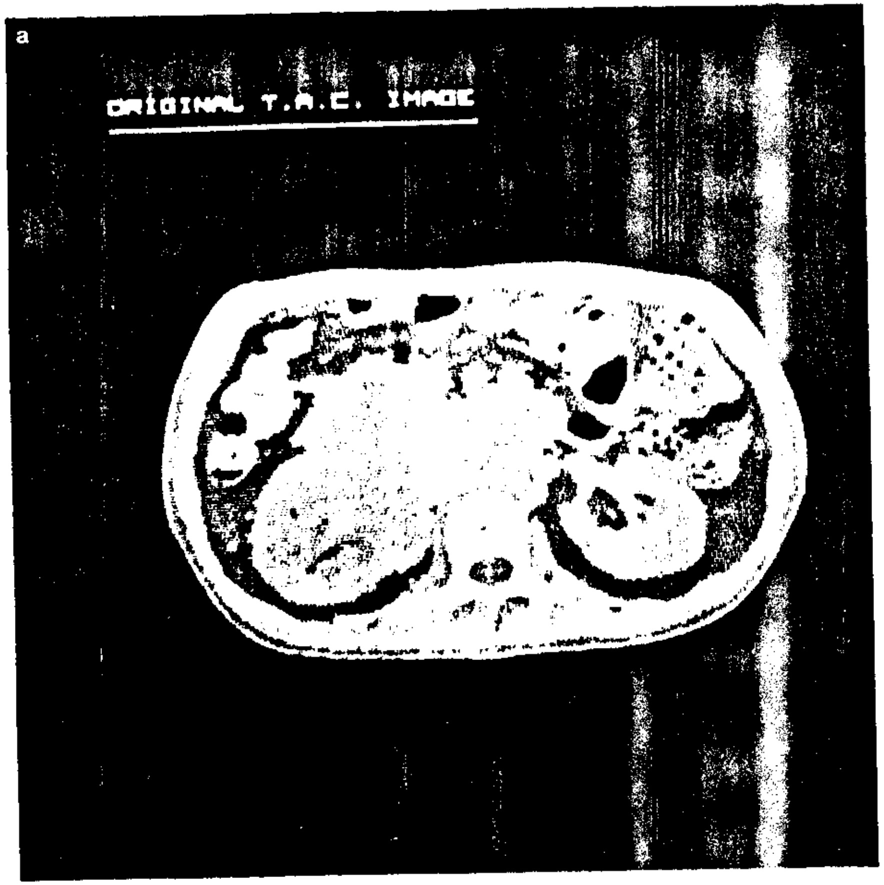


FIG. 24. Example of processing a computer tomography image: (a) original image; (b) result of linear stretching; (c) result of 2D FIR digital filtering of the parabolic type.

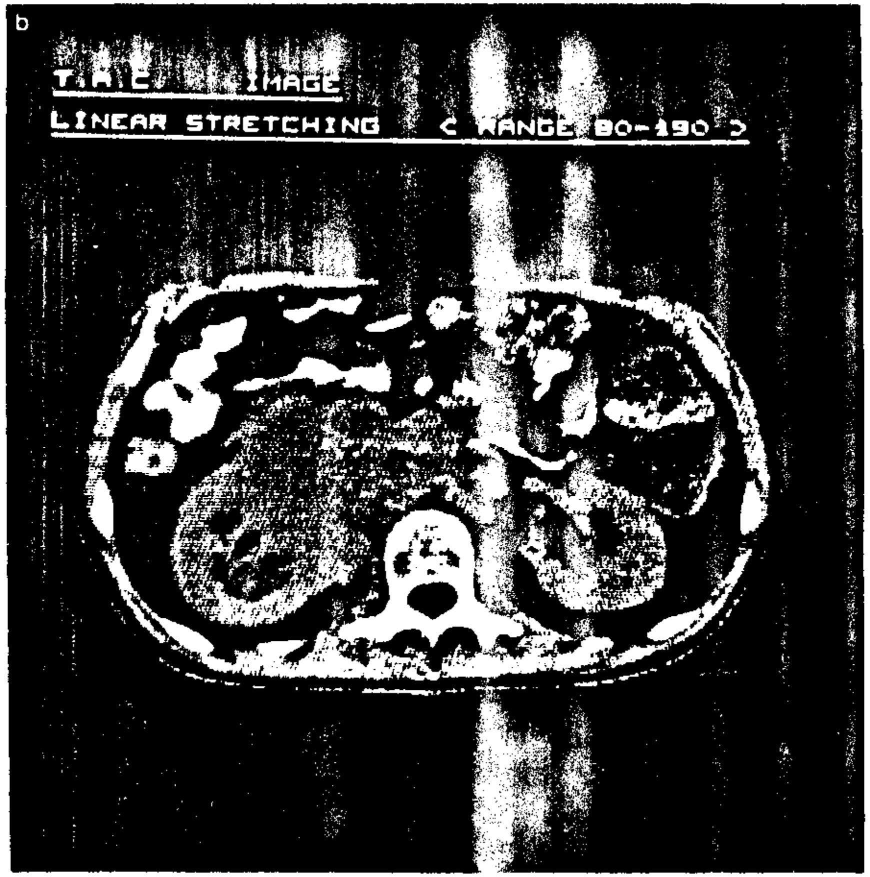


FIG. 24b

The first example regards the analysis of complex objects, where the goal is to process images taken of the objects and to produce an automatic object decomposition and subpart identification classification. The processing procedure is outlined in Fig. 25: by using a TV camera images are acquired in 3 colors (R,G,B—red, green, blue), then prefiltering is performed to reduce the noise; after boundary extraction, decomposition and syntactical analysis are performed. Figure 26 shows an example of the application of this procedure: in 26a an original digitized image (red color presented in black and white) is given representing a circuit board (acquisition with strong noise); in 26b the result of decomposition of the circuit board, obtained through a nonlinear filter of the type presented in Section III,A, an edge detector, and homogeneity operator applied on the three R,G,B images (Cappellini *et al.*, 1984b).

Another example regards the recognition and tracking of moving objects as on a transporting tape. The processing steps are the following: preprocessing with a nonlinear smoother [as in Eq. (26)]; edge detection (i.e., Sobel-type operator); spike elimination with a nonlinear operator [as in Eqs. (27) and (28)]; segmentation; object recognition by performing the FFT on the boundary of the object (distances of the boundary points from the centroid). An example of the application of this procedure is given in Fig. 27 on some mechanical objects. At the left there are the input digitized images of two

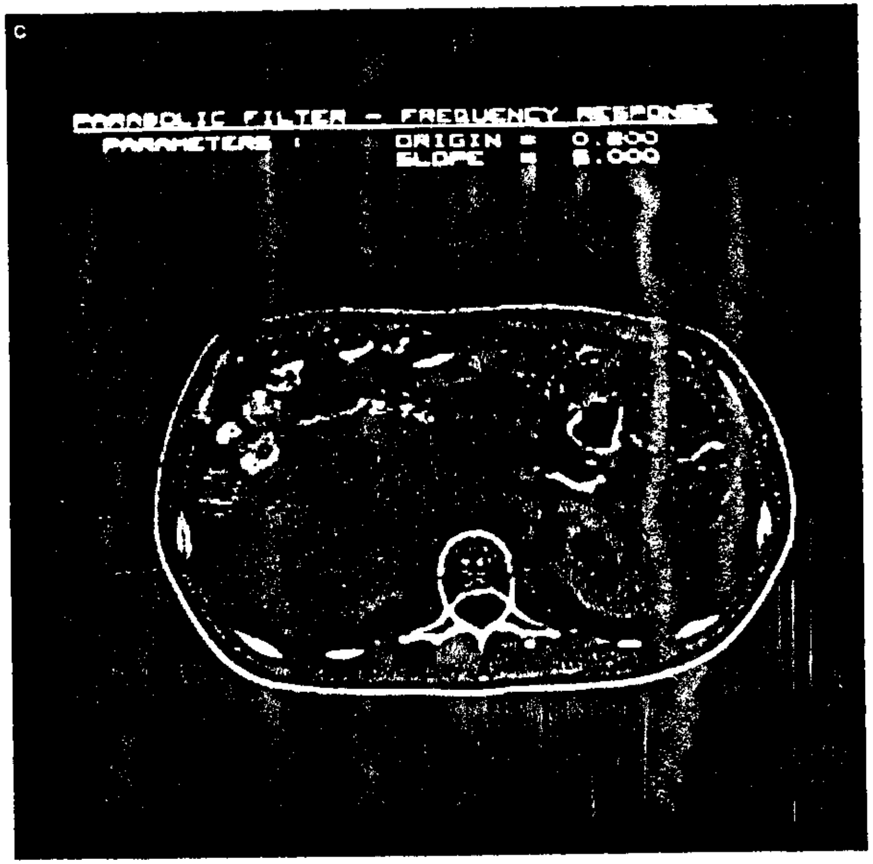

c
PARABOLIC FILTER - FREQUENCY RESPONSE
PARAMETERS : ORIGIN = 0.800
SLOPE = 8.000

FIG. 24c

positions; at the right the recognized objects are shown (each identified with a different color, here appearing as a different grey level) with perfect tracking of their movements (Cappellini and Del Bimbo, 1983).

With reference also to the above examples, in these robotics applications the preprocessing step is indeed very important (fast and efficient nonlinear filtering operators and edge detectors are required), in such a way as to reduce the noise and disturbances and extract the significant data in compressed form (that is limited to the really significant ones) for the best performance of the final recognition–classification algorithms and procedures.

TABLE I

EXPERIMENTAL RESULTS OBTAINED APPLYING DATA COMPRESSION USING TWO-DIMENSIONAL FFT AND FWT TRANSFORMATIONS TO NUCLEAR MEDICINE IMAGES WITH OR WITHOUT A TWO-DIMENSIONAL LOW-PASS DIGITAL PREFILTERING

	Compression ratio $C_{ra} = 2.5$ ($C_{ra} = 6$ with pre-filtering)		Compression ratio $C_{ra} = 3.5$ ($C_{ra} = 8$ with pre-filtering)	
	e_p	e_r	e_p	e_r
FFT	0.1322	0.0134	0.1581	0.0164
FWT	0.0483	0.0074	0.1088	0.0160

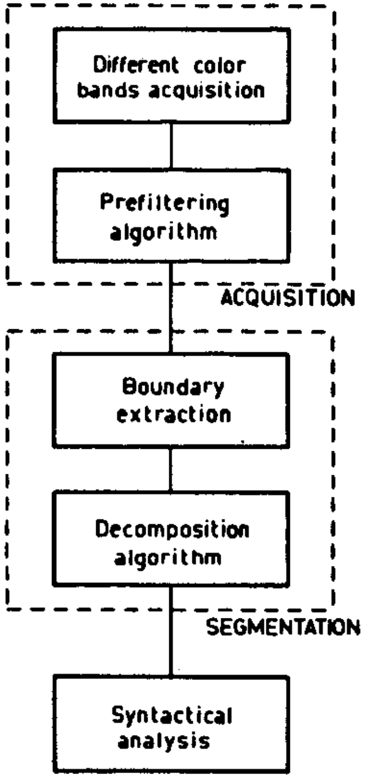


FIG. 25. Processing procedure for automatic object decomposition and subpart identification classification.

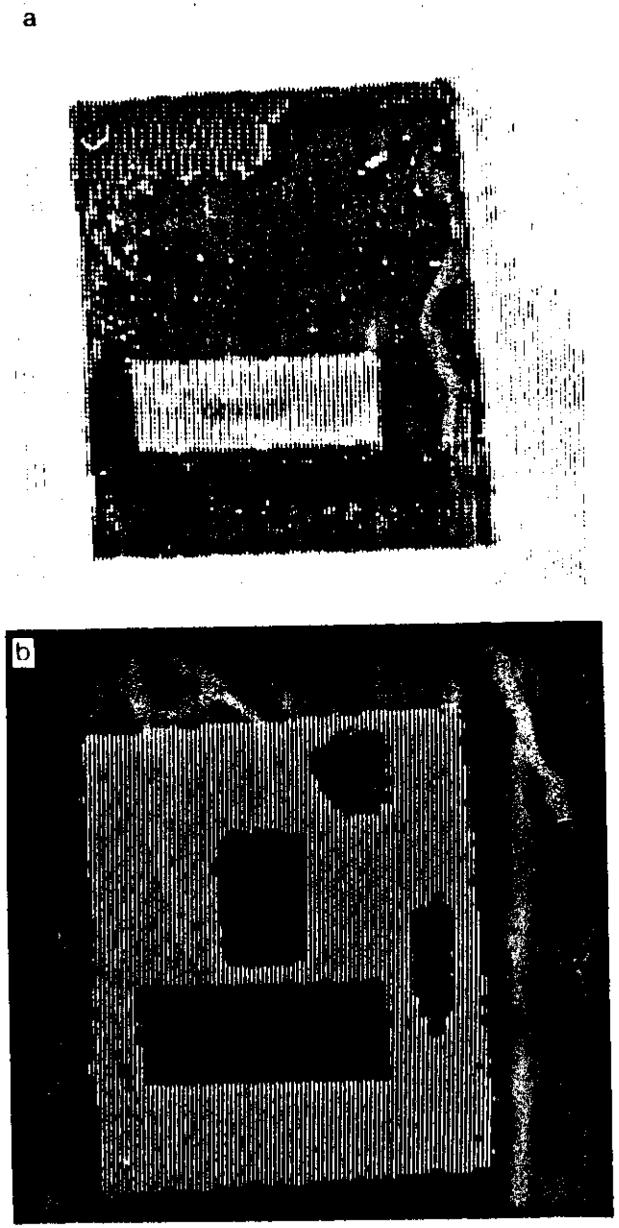


FIG. 26. Example of the application of the procedure in Fig. 25: (a) original digitized image representing a circuit board; (b) result of decomposition obtained through a nonlinear operator, an edge detector, and a homogeneity operator.

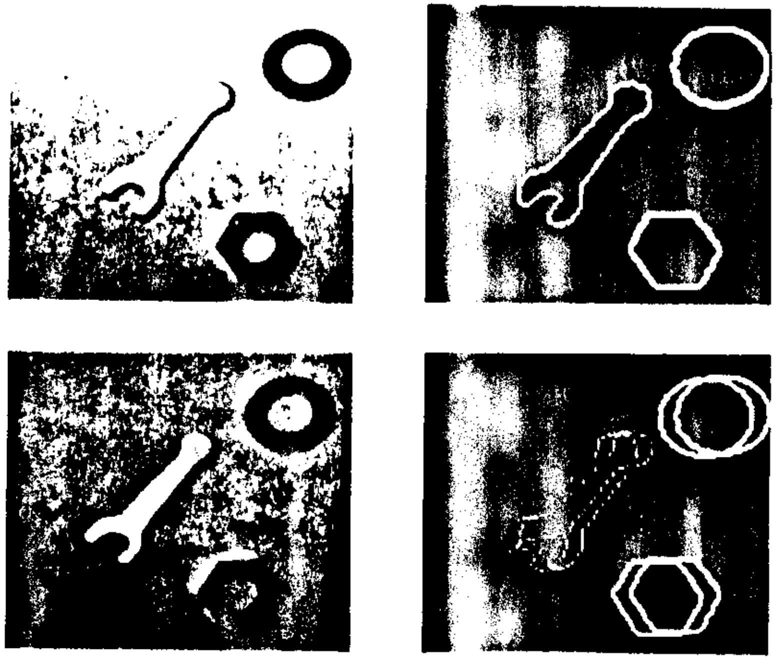


FIG. 27. Example of processing images related to moving objects: at left, the input digitized images; at right, the recognized objects with movement tracking.

REFERENCES

Abramson, N. (1963). "Information Theory and Coding." McGraw-Hill, New York.

Benelli, G., Bianciardi, C., Cappellini, V., and Del Re, E. (1977). *Proc. EUROCON–Euro Conf. Electrotech. Venice.*

Benelli, G., Cappellini, V., and Lotti, F. (1980). *Radio Electron. Eng.* **50**, 29.

Benelli, G., Cappellini, V., and Del Re, E. (1984). *IEEE Select. Areas Comm.* SAC-**2**, 77.

Berger, T. (1971). "Rate Distortion Theory—A Mathematical Basis for Data Compression." Prentice-Hall, New York.

Bernabò, M., Cappellini, V., and Emiliani, P. L. (1976). *Electron. Lett.* **12**, 288.

Brofferio, S., Cafforio, C., Rocca, F., and Ruffino, U. (1975). *Proc. Florence Conf. Digital Signal Process.* p. 158.

Calzini, M., Cappellini, V., and Emiliani, P. L. (1975). *Alta Frequenza* **44**, 747.

Cappellini, V. (1979a). *Proc. JUREMA Conf., Zagreb.*

Cappellini, V. (1979b). *Proc. Int. Workshop Image Process. Astron., Trieste* p. 258.

Cappellini, V. (1980). *Int. Remote Sensing*, **1**, 175.

Cappellini, V. (1983). *Proc. IEEE Int. Symp. Circuits Systems, Newport Beach* p. 402.

Cappellini, V. (1984). *Proc. EARSeL/ESA Symp. Integrated Approaches Remote Sensing, Guildford* p. 325.

Cappellini, V., and Del Bimbo, A. (1983). *In* "Issues in Acoustic Signal/Image Processing and Recognition" (C. H. Chen, ed.), p. 283. Springer-Verlag, Berlin and New York.

Cappellini, V., and Emiliani, P. L. (1983). *Proc. MEDINFO-83, Amsterdam* p. 682.

Cappellini, V., and Odorico, L. (1981). *Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Atlanta* p. 1129.

Cappellini, V., Chini, A., and Lotti, F. (1976). *Proc. Int. Techn. Scie. Meet. Space, Rome* p. 33.

Cappellini, V., Constantinides, A. G., and Emiliani, P. (1978). "Digital Filters and Their Applications." Academic Press, New York.

Cappellini, V., Carlà, R., Conese, C., Maracchi, G. P., and Miglietta, F. (1984a). *Proc. EARSeL/ESA Symp. Integrated Approaches Remote Sensing, Guildford* p. 23.

Cappellini, V., Del Bimbo, A., and Mecocci, A. (1984b). *Image Vision Comput.* **2,** 109.

Costa, J. M., and Venetsanopoulos, A. N. (1974). *IEEE Trans. Acoust. Speech Signal Process.* **ASSP-22,** 432.

Dudgeon, D. E. (1975). *IEEE Trans. Acoust. Speech Signal Signal Process.* **ASSP-23,** 242.

Ekstrom, M. P. (1980). *IEEE Trans. Acoust. Speech Signal Process.* **ASSP-28,** 16.

Harris, O. B., and Mersereau, R. M. (1977). *IEEE Trans. Acoust. Speech Signal Process.* **ASSP-25,** 492.

Hilberg, W., and Rothe, P. G. (1971). *Inf. Control* **18,** 103.

Hu, J. V., and Rabiner, L. R. (1972). *IEEE Trans. Audio Electroacoust.* **AU-20,** 249.

Kaiser, J. F. (1966). *In* "System Analysis by Digital Computer" (F. F. Kuo and J. F. Kaiser, eds.), p. 218. Wiley, New York.

McClellan, J. H. (1973). *Proc. Annu. Princeton Conf. Inf. Sci. Systems, 7th,* p. 247.

Maria, G. A., and Fahmy, M. M. (1974). *IEEE Trans. Acoust. Speech Signal Process.* **ASSP-22,** 16.

Mecklenbrauker, W. F. G., and Mersereau, R. M. (1976). *IEEE Trans. Circuits Systems* **CAS-23,** 414.

Mersereau, R. M., and Dudgeon, D. E. (1975). *IEEE Proc.* **63,** 610.

Mersereau, R. M., Mecklenbrauker, W. F. G., and Quatieri, T. F., Jr. (1976). *IEEE Trans. Circuits Systems* **CAS-23,** 405.

Oppenheim, A. V., and Shafer, R. W. (1975). "Digital Signal Processing." Prentice-Hall, New York.

Pratt, W. K. (1978). "Digital Image Processing." Wiley, New York.

Prosperi, L. (1983). Thesis, Department Electrical Engineering, University of Florence.

Shannon, C. E. (1959). *IRE Natl. Conv. Rec.* **7,** 142.

Shannon, C. E., and Weather, W. (1949). "The Mathematical Theory of Communication." Univ. of Illinois Press. Urbana.

Shanks, J. L., Treitel, S., and Justice, J. H. (1972). *IEEE Trans. Audio Electroacoust.* **AU-20,** 115.